Рачунарство

Computing and information engineering

ESXI, Microsoft Hyper-V, XenServer: FileSystem Performance Comparasion for Type-1 Hypervisors

Borislav Đorđević, *Member, IEEE*, Valentina Timčenko, *Member, IEEE*, Stefan Kovačević

***Abstract*—This author guide gives information and a template for preparing camera-ready manuscripts for the IcETRAN Conference. You may use the built-in styles from this document if you are using Microsoft *Word* XP or later. Otherwise, use this document as an instruction set. Paper titles should be written in uppercase and lowercase letters, not all uppercase. Avoid using subscripts, superscripts or special symbols in the title and in the abstract. Full names of authors are required in the author field, not only the initials of given names. Define all abbreviations in the abstract when first using them. Do not cite references in the abstract. Do not delete the 2pt blank line immediately above the abstract; it sets the footnote at the bottom of the left column on the first page containing author affiliations and email addresses.**

***Index Terms*—Enter key words or phrases, separated by semicolons.**

I. Introduction

The application of the virtualization technologies can provide enterprises high advantages in terms of saving resources and providing additional security. It achieves optimal hardware utilization, stronger reliability and security improvements, and simpler administration. A hypervisor is a software abstraction layer that lies between hardware and operating system [1].

There are two types of hypervisors (Figure 1): type-1 hypervisor that is executed directly on hardware and manages guest operating systems (Hyper-V, ESXi, Xen); and type-2 hypervisor that is executed on the host operating system (VirtualBox, VMware Workstation).

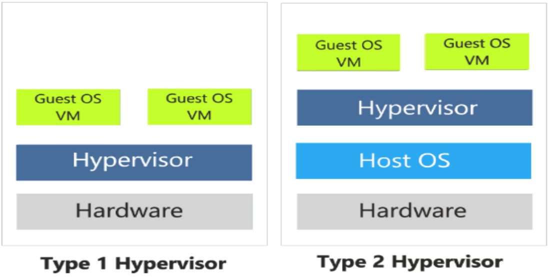


Figure 1. Hypervisor types and differences.

As the type-1 hypervisor has direct access to hardware, while type-2 hypervisor accesses hardware through host operating system, we assume that type-1 hypervisor provides more scalability, reliability, and better performance [2].

Type-1 hypervisors are grouped into two subcategories: Monolithic and Micro-kernelized, where the main difference is reflected through device drivers characteristics (i.e. the drivers location). For the needs of this research we have evaluated three type-1 hypervisor representatives: Microsoft Hyper-V (hereinafter referred to as Hyper-V), VMware ESXi (hereinafter reffered as ESXi), and Xen with the Windows Server 2012 R2 guest operating system.

II. Related work, objective and motivation

In this paper, the research is focused to the performance comparison of three type-1 hypervisors. Numerous studies refer to the research results related to Hyper-V, ESXi, Xen, and KVM, mainly basing the results on the use of proven benchmarks: FileBench, Bonnie, HDTunePro, ATTO [12], [13].

Since the development in this area is still on the run, it is not a surprise to find large number of papers refering to the comparison and performance evaluation of different virtualization approaches. The main focus is on the speed of data writing and reading (I/O), which is particularly important for applications running in cloud environment. Additionaly, many studies explore the solutions for virtualization management, I/O speed, security, capabilities, etc [14].

The purpose of this work is to compare the performance of write and read operations when using Hyper-V, ESXi, and Xen on identical hardware, the same virtual machine parameters and the same guest operating system (Windows Server 2012 R2). Analized hypervisors are representatives of type-1, and the fundamental difference is reflected in the fact that Hyper-V uses paravirtualization, while ESXi and Xen apply the full hardware virtualization. ATTO Benchmark 4.01. software is used to determine input and read performance on virtual machines. A particular set of hypotheses and mathematical model are defined, based on which the performance is measured and interpreted [4], [13].

III. Hypotheses of the expected behavior

All hypervisors are type-1, and work directly on the

Borislav Đorđević - Institute Mihajlo Pupin, Volgina 15, 11000 Belgrade, Serbia, (borislav.djordjevic@pupin.rs)
Valentina Timčenko - Institute Mihajlo Pupin, Volgina 15, 11000 Belgrade, Serbia (valentina.timcenko@pupin.rs)
Stefan Kovačević - VISER, School of Electrical and Computer Enginering of Applied Studies, Belgrade, Serbia (kovacevic.stefan@hotmail.com)

hardware. They are very thin and realized in microkernel architecture. The total processing time for each workload T_W can be calcoulated according to the following equation:

$$T_W = f((T_{RR}, T_{SR}, T_{RW}, T_{SW}) \qquad (1)$$

where T_{RR} and T_{SR} represent random and sequential read components, and T_{RW} and T_{SW} stand for the random and sequential write components. For each specific workload we have calculated the expected access time for a filesystem (FS) that includes five components, as shown below:

$$T_{WL} = f(\ T_D + T_M + T_{FL} + T_{FB} + T_J + T_{HK}) \qquad (2)$$

where T_{WL} represents the total time to complete all operations for the defined workload. T_D, T_M, T_{FL}, T_{FB}, T_J, T_{HK} represent the time required to complete all operations related to the directory, metadata, free lists, file blocks, journaling and house-keeping operations in the FS.

In the context of virtual environment, there are three components that impact the T_W:

$$T_W = f(\text{gFS-proc, H-proc, hFS-proc}) \qquad (3)$$

1) *gFS-proc:* guest FileSystem processing component represent the benchmark interaction with guest filesystem. For all the explored hypervisors, as the test environment relies on the use of the identical benchmark, VMs, and guest filesystem (NTFS), it is expected that this component provides an identical effect on T_w.

2) *(H-proc):* hypervisor processing component is different for explored ESXi, Hyper-V and Xen and MS Windows OS based guests hypervisors. It depends on the type of virtualization and hypervisor delays.
When considering the context of virtualization type:

a) ESXi: For most guest operating system, Xen employs only the full-hardware virtualization. It is based on the CPU assisted full-hardware WMware virtualization.

b) Xen: exhibits strong paravirtualization effects for Linux PV guests. In the case of the MS Windows guests, Xen can only use the full-hardware virtualization. It is QEMU full-hardware virtualization with CPU hardware assisted virtualisation (HVM guests).

c) Hyper-V: For MS Windows based guests, it manifests strong paravirtualization effects, relying on VM bus (RAM) and complete Hyper-V vitrualization stack componets.

We expect remarkable advantage for Hyper-V, because paravirtualization should be solidly faster than full-hardware virtualization. Also Paravirtualization is less portable and compatible compared to full-hardware virtualization.

In the context of the Hypervisor-processing type: different delays depanding on the used hypervisor: ESXi, Xen and Hyper-V. Delay represents the time required for the hypervisor to receive the requests from virtual hardware of guest OS and forward them to the host OS drivers. The FS requests from the guest filesystem are forwarded to the FS host filesystem. In this case, we expected remarkable advantage for Hyper-V, mostly due to paravirtualization and smaller number of context switches.

3) *hFS-proc:* it is expected that the Host FileSystem processing component generates big difference between analyzed hypervisors. ESXi and Xen rely on ext4 as host OS filesystem, while Hyper-V uses NTFS. Both filesystem are modern and 64bit, but with performance differences.

As the tests are focused on the performance of MS Windows guest, it is expected the dominant influence of the 2nd and 3rd component from the formula (3), especially 2nd component which depends on the virtualization type.

This research is focused to the use of the same gostOS. It will be used as a native system and excuted as full hardware emulation on ESXi and Xen, while Hyper-V will paravirtualize the same guestOS. Based on the defined hypothesis and using practical experience, Hyper-V is expected to produce the best performance.

IV. Microsoft Hyper-v, VMware Esxi and XenServer

Microsoft Hyper-V (Figure 2) is a native hypervisor that runs directly on the hardware, just below the operating system. The virtualization software runs in the "Parent partition" and has direct access to the hardware devices. For the needs of freeing up the space for the guest operating system, the "Parent partition" creates new "child partitions" that do not have direct access to hardware, but represent virtual devices/resources. VM Bus provides communication between partitions. The parent partition starts the Virtualization Service Provider (VSP), which connect to the VMBus and processes the requests of the child partition to access devices. Virtual devices assigned to a child partition run a (VSC) , which redirects requests to VSPs in the parent partition via VMBUS. VCSs are drivers in virtual machines, which together with other component provide advanced performance and features for a virtual machine [3], [15].

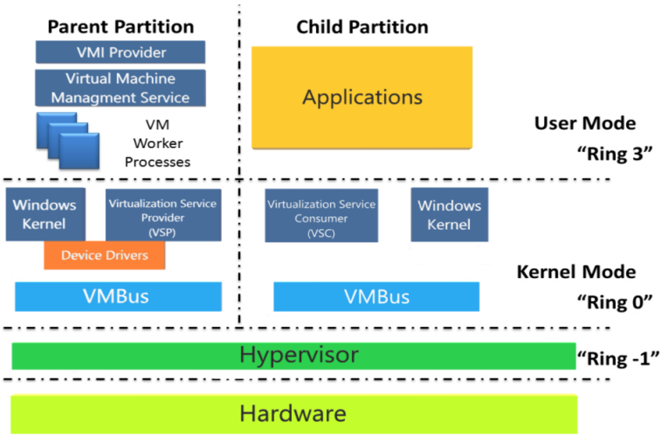


Figure 2. Microsoft Hyper-V VM Bus.

For each guest OS, Hyper-V creates a single virtual hard disk format as .vhdx file (older format .vhd). It uses VT-x or AMD-v hardware acceleration. Given that Hyper-V is the native hypervisor, as long as it is installed, other software cannot use Vt-x or AMD-v [4].

Hyper-V is a micro-core hypervisor that uses para-virtualization and full-virtualization [5], while hardware drivers do not belong to the hypervisor layer. Para-virtualization is more secure than the full-virtualization.

ESXi (Figure 3) is a native hypervisor. It is not an application software installed on the operating system, but a virtualization software that runs the kernel. Monolithic hypervisors use hardware emulation (full virtualization) [6]. The monolithic hypervisor manages hardware access to every virtual machine. It contains all the hardware drivers that virtual machines need for proper functioning (storage devices, network). The advantage of this design is that it does not require a host operating system but the hypervisor acts as an operating system platform.

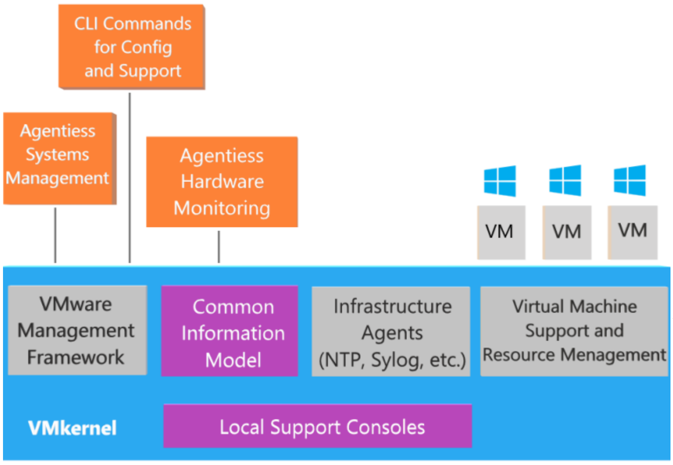

CLI Commands for Config and Support
Agentiess Systems Management
Agentiess Hardware Monitoring
VM
VM
VM
VMware Management Framework
Common Information Model
Infrastructure Agents (NTP, Sylog, etc.)
Virtual Machine Support and Resource Menagement
VMkernel
Local Support Consoles

Figure 3. ESXi architecture.

VMkernel is responsible for manages virtual machines and also manages access to basic physical hardware.

The main processes that run at the top of the kernel are: [7]

- **DCUI -** Interface for the low-level configuration and management, primarily used for the initial basic configuration
- **VMM** - each running machine has its own VMM and VMX process, which provides an executable environment for the virtual machine.
- **CIM** - allows you to manage hardware from remote applications via API.

Xen (Figure 4) is the only native hypervisor that is available as open source [8]. Xen hypervisor is a software layer that runs direcly on the hardware. It is also responsible for CPU scheduling, as well as memory allocation for virtual machines. When running Xen, the hypervisor takes control of the system and then loads the first Dom0 guest operating system. Dom0 is a modified Linux kernel that has special access rights to physical I/O resources, as well as the right to interact with other virtual machines (Dom) [9].

The Dom, unlike Dom0, does not have direct access to the hardware. DomU PVS are modified Linux, Solaris, and FreeBSD operating systems, while DomU HVM guests run the standard Windows operating system or any other immutable operating system [10].

In Hyper-V, hardware drivers should not be part of the hypervisor layer.

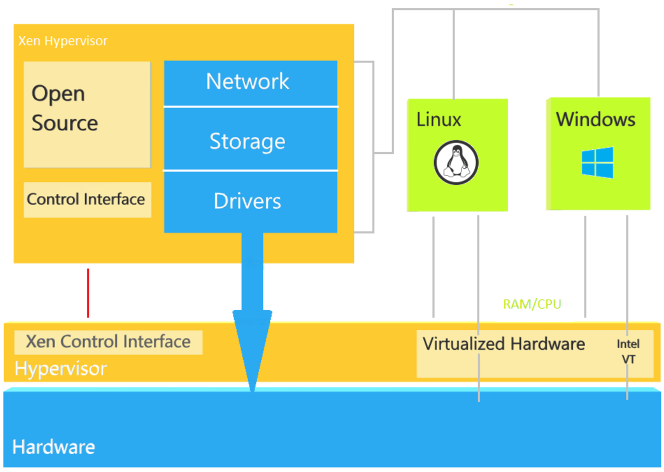


Figure 4. Xen architecture

V. Testing

The presented graphs are based on the average values from three conducted tests run for each virtual machine. In order to achieve equal testing for all participants, the same hardware configuration was used. Table 1 shows the hardware configuration with the components used for the tests.

TABLE I
SERVER TEST ENVIRONMEENT

PHE ProLiand DL360 G5	
Component	***Characteristic***
CPU	2 x Intel Xeon E5420 QuadCore 2.5GHz
RAM	48GB DDR2
Storage Controllers	Smart Array p400i 256MB
SSD	Geil Zenith R3 GZ25R3 128GB
Network	2 x 1Gb/s

The parameters of virtual machines are shown in Table 2. All used virtual machines have identical characteristics.

TABLE II
VIRTUAL MACHINE PARAMETERS

Component	***Characteristic***
vCPU	4
RAM	4GB
Disk	20GB + 5GB
OS	Windows Server 2012 R2

Atto Disk Benchmark 4.0 is used for test procedures. It is designed to measure the performance of storage systems with different data transmission sizes and lengths of read and write tests. It supports 512B to 64MB data transfer size and 64KB to 32GB transfer lengths.

Testing is performed on the same hardware, with each virtual machine having an identical environment. The impact of other system operations is minimized during testing. First, we have installed Windows Server with Hyper-V, applied the configuration for virtual machine and installed the Windows Server 2012 as a guest. After running the testing procedure, the disk was formatted. Then, the VMware ESXi is installed, the virtual machine configured, and the corresponding guest OS installed. The tests were performed three times for each test to achieve data reliability. Next, the ESXi disk is reformatted and XenServer installed. We have configured VM, installed the appropriate operating system and run the performance measurement tests. Figure 5 and Table 3 show the write speeds with configuration based on the use of one virtual machine.

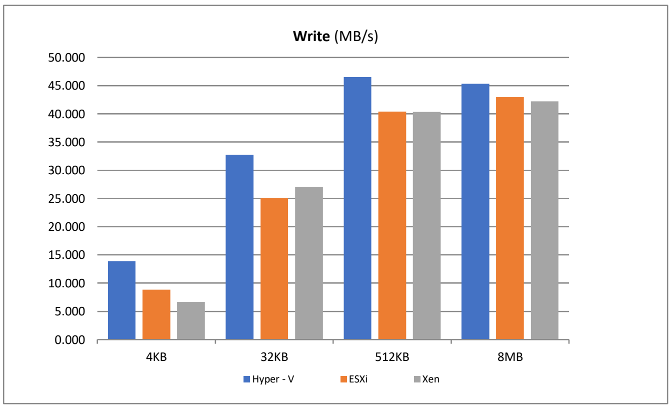


Figure 5. Write performance for a single virtual machine. (in MB/s)

TABLE III
WRITE PERFORMANCE FOR A SINGLE VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	13.89 MB/s	32.73 MB/s	46.54 MB/s	45.31 MB/s
ESXi	8.84 MB/s	25.05 MB/s	40.38 MB/s	42.97 MB/s
XenServer	6.66 MB/s	27.05 MB/s	40.33 MB/s	42.24 MB/s

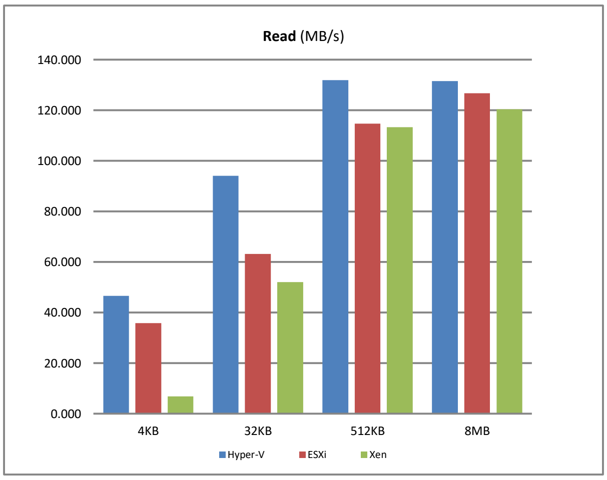


Figure 6. Read performance for a single virtual machine. (in MB/s)

Figure 6 and Table 4 show read speeds for single virtual machine case.

TABLE IV
READ PERFORMANCE FOR FOR A SINGLE VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	46.6 MB/s	94.12 MB/s	131.91 MB/s	131.59 MB/s
ESXi	35.92 MB/s	63.20 MB/s	114.66 MB/s	126.74 MB/s
XenServer	6.92 MB/s	52.05 MB/s	113.29 MB/s	120.37 MB/s

In the case for writing operation, Hyper-V shows the best performance for all block sizes. ESXi shows better writing performance than Xen with 4KB and 8MB block sizes, while Xen shows better writing performances than ESXi when operating with 32KB block size. In the case of the 512KB block size, ESXi and Xen show similar performance. For the case of read operation, Hyper-V continues to be dominantly the best. ESXi shows better read performance when compared to Xen (for all block sizes), while Xen has the strongest performance deviation for the case of writing 4KB (small block sizes).

In the second part of the test, two virtual machines were run simultaneously. The results of the write speed are presented in figure 7 and Table 5.

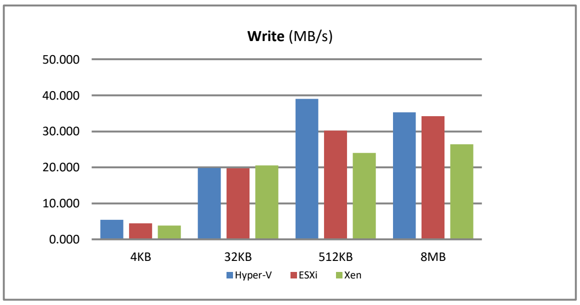


Figure 7. Write performance for two virtual machine. (in MB/)

TABLE V
WRITE PERFORMANCE FOR FOR TWO VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	5.42 MB/s	19.81 MB/s	39.01 MB/s	35.31 MB/s
ESXi	4.47 MB/s	19.77 MB/s	30.19 MB/s	34.55 MB/s
XenServer	3.80 MB/s	20.51 MB/s	24.01 MB/s	26.44 MB/s

The read speed results are shown in Figure 8 and Table 6.

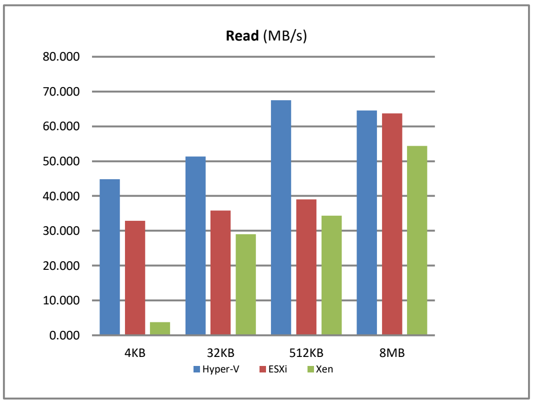


Figure 8. Read performance for two virtual machine. (in MB/s)

TABLE VI
READ PERFORMANCE FOR FOR TWO VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	44.82 MB/s	51.32 MB/s	67.51 MB/s	64.58 MB/s
ESXi	32.89 MB/s	35.89 MB/s	39.03 MB/s	63.73 MB/s
XenServer	3.82 MB/s	29.07 MB/s	34.32 MB/s	54.35 MB/s

Again, with 4KB, 512KB, and 8MB blocks size, Hyper-V provided the best results, while Xen was the best option when operating with 32KB blocks size. For write operation, in the case of 4KB, 512KB and 8MB block sizes, ESXi provides better performance when compared to Xen. In general, Hyper-V shows the best performance while Xen again has the highest deviation of write performances when opetaing with small blocks (4KB).

The third test relied on running 3 virtual machines at the same time, and it is repeated three times. The average results are shown in figure 9 and table 7.

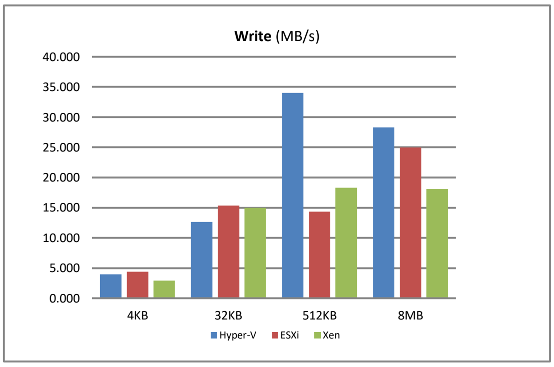


Figure 9. Write performance for three virtual machine. (in MB/s)

TABLE VII
WRITE PERFORMANCE FOR FOR THREE VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	3.98 MB/s	12.65 MB/s	34.01 MB/s	28.28 MB/s
ESXi	4.39 MB/s	15.32 MB/s	14.33 MB/s	24.98 MB/s
XenServer	2.94 MB/s	14.95 MB/s	18.32 MB/s	18.11 MB/s

The read speed results are shown in Figure 10 and Table 6.

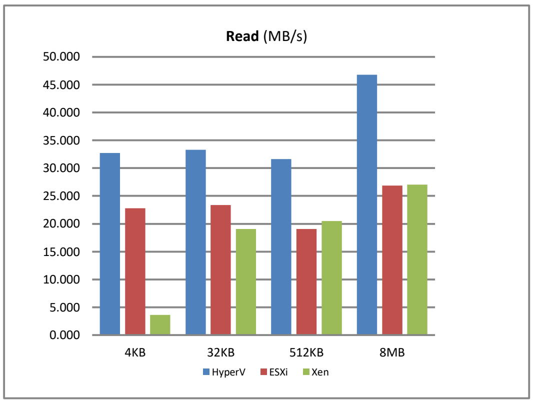


Figure 10. Read performance for three virtual machine. (in MB/s)

TABLE VIII
READ PERFORMANCE FOR FOR THREE VM

Atto Benchmark	4KB	32KB	512KB	8MB
Hyper-V	32.73 MB/s	33.31 MB/s	31.61 MB/s	46.78 MB/s
ESXi	22.78 MB/s	23.37 MB/s	19.06 MB/s	26.88 MB/s
XenServer	3.65 MB/s	19.09 MB/s	20.49 MB/s	27.05 MB/s

Unlike the previous two tests, in this test ESXi shows slightly better results for write operation for small block sizes (4KB and 32KB), while Hyper-V still dominates for block sizes of 512KB and 8MB. That Hyper-V dominance is poved for read operation as well.

In general, for all three tests routines Hyper-V with guest OS Windows Server 2012 has showed the best write and read results (Table IX). This is expected, mostly due to the paravirtualization of the guest OS, and is reflected through formula 3 and its 2nd and 3rd components. Hyper-V is in most of the cases much better than the other two hypervisors, mostly due to the effects of paravirtualization. It is mostly obvious in the cases of VM BUS.

TABLE IX
HYPERVISOR PERFORMANCE RESULTS

	1VM Write	1VM Read	2VM Write	2VM Read	3VM Write	3VM Read
Hyper-V	4KB, 32KB, 512KB, 8MB	4KB, 32KB, 512KB, 8MB	512KB, 8MB	4KB, 32KB, 512KB	512KB, 8MB	4KB, 32KB, 512KB, 8MB
ESXi			4KB	8MB	4KB, 32KB	
Xen			32KB			

Hyper –V is dominantly the best for:

- 1VM Write test - block sizes 4KB, 32KB, 512KB and 8MB
- 1VM Read test - block sizes 4KB, 32KB, 512KB and 8MB
- 2VM Write test - block sizes 512KB and 8MB
- 2VM Read test - block sizes 4KB, 32KB and 512KB
- 3VM Write test - block sizes 512KB and 8MB

- 3VM Read test - block sizes 4KB, 32KB, 512KB and 8MB.

Xen and ESXi results are relatively similar. ESXi was better for:

- 1VM Write test - block sizes 4KB and 8MB
- 1VM Read test - block sizes 4KB, 32KB, 512KB and 8MB
- 2VM Write test - block sizes 4KB, 512KB and 8MB
- 2VM Read test - block sizes 4KB, 32KB, 512KB and 8MB
- 3VM Write test - block sizes 4KB, 32KB and 8MB
- 3VM Read test - block sizes 4KB, 32KB, and 8MB.

Xen was better in the case of:

- 1VM Write test - block sizes 32KB
- 2VM Write test - block size 32KB
- 3VM Write test - block size 512KB
- 3VM Read test - block sizes 512KB and 8MB.

VI. Conclusion

In this paper we compared three powerful hypervisors of type-1: Hyper-V, ESXi, Xen, based on the use of Windows Server 2012 R2 as the guest operating system. We have set up mathematical model, measured the performances, presented and interpreted the obtained results based on the defined mathematical model. In majority of the tested cases Hyper-V proved to be significantly better than other two tested hypervisors, basicaly due to the use of paravirtualization. ESXi and Xen proved to be similar, with some variations depending on the analzyed case. The reason for this lies in the solution implemented by these two hypervisors, enforcing the full virtualization. ESXi uses its own solution, while Xen implements an open source QEMU solution. Still, in some cases these two hypervisors showed better performance than the favored Hyper-V. Future work will include a comparison of these three hypervisors with Linux-based guests.

Acknowledgment

The work presented in this paper has partially been funded by the Ministry of Education, Science and Technological Development of the Republic of Serbia: V. Timcenko by grants TR-32037, TR -32025, and B. Djordjevic by grant III-43002.

References

[1] Fatma Bazargan, Chan Yeob Yeun, Mohamed Jamal Zemerly, State-of-the-Art of Virtualization, its Security Threats and Deployment Models, international Journal for Information Security Research (IJISR),Volume 2, Issues 3/4, September/December 2012.

[2] VMWare,“Understanding Full virtualization, Para virtualization and hardware Assist“ 2007.

[3] Microsoft,“Hyper-V Architecture“,online:https://msdn.microsoft.com/en-us/library/cc768520(v=bts.10).aspx, 2017.

[4] Yrewicze, Andry,“VMware vs Hyper-V: Architectural Differences", 2018

[5] A. Finn and P. Lownds, Mastering Hyper-V Deployment, Wiley Publishing Inc.

[6] John Savill, What’s the difference between monolithic and microkernelized hypervisors?, http://windowsitpro.com/virtualization/q-what-s-difference-between-monolithic-and-microkernelized-hypervisors, 2008

[7] VMware, "The Architecture of VMware ESXi", 2007.

[8] Linux Foundation, “The Xen Project, the powerful open source industry standard for virtualization”

[9] Linux Foundation, “How Xen Works”

[10] Hwanju K., Heeseung J., and Joonwon L. XHive: Efficient Cooperative Caching for Virtual Machines , IEEE Transactions on Computers, 2011.

[11] Atto Disk Benchmark, onlin:https://www.atto.com/disk-benchmark/

[12] Borislav Đorđević, Nemanja Maček, Valentina Timčenko, Performance Issues in Cloud Computing:KVM Hypervisor’s Cache Modes Evaluation, Vol. 12, No. 4, 2015, pp 147-165, http://uni-obuda.hu/journal/Dordevic_Macek_Timcenko_60.pdf,

[13] Varun Kumar Manik, Deepak Arora, "Performance comparison of commercial VMM: ESXi, Xen, Hyper-V & KVM", 2016 https://ieeexplore.ieee.org/abstract/document/7724572

[14] D. Armstrong, K. Djemame: Performance Issues in Clouds: An Evaluation of Virtual Image Propagation and I/O Paravirtualization. The Computer Journal, Vol. 54, No. 6, 2011, pp. 836-849.

[15] Leandro Carvalho, Windows Server 2012 Hyper-V Cookbook, Packt Publising, 2012.

Example of Google Cloud Speech API usage and interpretation of recognized content on the E-ink display

Igor Medenica, Nikola Radulović, Vladimir Milovanović, Jelena Vasiljević and Miloš Jovanović

Abstract— **Accelerated lifestyle and increasing technological development requires an ever faster way to establish communication between two people. Virtual communication has become a common and frequent way of contact between people in the 21st century. It is performed via the Internet, voice or SMS messages. This paper presents one of the custom types of accelerated communication. Using the Cloud Speech API, speech recognition and conversion to text format is performed. The recognized content from the Android device is transmitted via Bluetooth communication to the Arduino UNO to which the E-ink display is connected, on which the received content is displayed. The use of this system speeds up communication with people who are unable to communicate in any other way, while maintaining privacy. The advantages of use are portability and small dimensions, which enable easy viewing of content without the use of a mobile device.**

Index Terms— **Google Cloud Speech to Text, Waveshare e-Paper display, Arduino UNO, HC-05**

I. Introduction

The development of information and communication technologies contributes to the introduction of numerous innovations that facilitate and accelerate the establishment of contact between two or more persons. Smartphones contribute to the reduction of direct communication and increase the text communication that is realized via SMS messages or via the Internet. Voice over text (Automatic Speech Recognition, ASR) is becoming a very popular method that is increasingly used in practice.

Voice recognition has been brought to a level of perfection with a small percentage of error. The development of artificial intelligence enables a large number of ASR systems that provide high quality transcripts. Quite a large number of successful technology companies have created such systems for both research and improvement of technological services. Some of the famous ones are Google Cloud, IBM Watson, Microsoft Azure.

Research has shown that the quality of transcripts reaches the same level as manual transcripts [1]. This type of technology supports the daily life of people with hearing impairments. In people with this disability, textual communication via smart devices is present in a large percentage [2]. Feedback on whether a message has arrived or not, whether a person is able to respond and much more information that can replace a direct form of communication plays an important role [3]. Many of these services strive to customize tools to use as many spoken languages as possible. The distinction between ASR systems can be divided into systems that recognize isolated words and those that recognize related isolated words, and a large number of systems are distinguished by the size of the dictionary (the amount of words which can be recognized) [4]. The Google Cloud Speech-to-Text API is used to make the problem solving more successful.

In addition to all the advantages of this type of system, one of its main disadvantages is portability, storage space and battery life [5]. This system tries to show that with the selected hardware and the use of free Google services, these problems can be avoided.

II. Design of the system

To implement the presented system, we used an Android phone, Arduino UNO programming board, HC-05 Bluetooth module and Waveshare e-Paper display. The system can be grouped into two entities (Fig. 1):

- speech recognition - consists of an Android application that communicates with the Google Cloud platform
- data presentation - consists of an Arduino Uno board with a Bluetooth module and an E-ink display

The user starts a conversation session via an application on a smartphone with the Android operating system. User speech is recognized using the Google Cloud API. The mentioned platform converts the recognized audio sample into text format in real time. Text content is forwarded via Bluetooth communication to an Arduino Uno board that prints the transcript on an e-Paper display.

Igor Medenica is with the School of Computing, University UNION, 6 Knez Mihajlova, 11020 Belgrade, Serbia (e-mail: medenicaigor@gmail.com).

Nikola Radulović is with the Faculty of Science, University of Novi Sad, Trg Dositeja Obradovica 3, 21000 Novi Sad, Serbia (e-mail: nikolardlvc@gmail.com).

Vladimir Milovanvić is with the School of Computing, University UNION, 6 Knez Mihajlova, 11000 Belgrade, Serbia (e-mail: vmilovanovic@raf.rs).

Jelena Vasiljević is with the School of Computing, University UNION, 6 Knez Mihajlova, 11000 Belgrade, Serbia (e-mail: jvasiljevic@raf.rs).

Miloš Jovanović is with the School of Computing, University UNION, 6 Knez Mihajlova, 11000 Belgrade, Serbia (e-mail: mjovanovic@raf.rs).

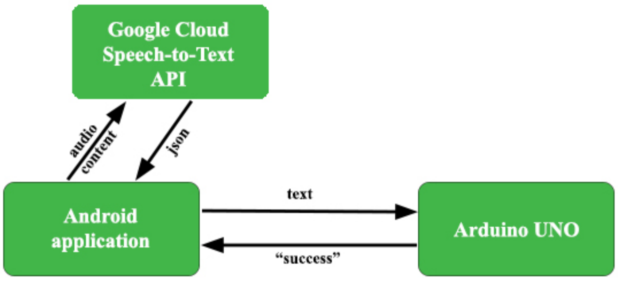


Fig. 1. System structure and communication between entities

A. *Hardware Infrastructure*

The paper used a smartphone with Android operating system version 9.0. The features of the phone are 2 GB of RAM and it has a CPU with four cores.

The Arduino Uno is an ATmega328P based microcontroller board. It has 14 digital input / output pins (6 of which can be used as PVM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB port, a power port, an ICSP header and a reset button.

The HC-05 is one of the types of Bluetooth modules that has two-way wireless functionality. This module can be used to communicate between two microcontrollers such as an Arduino or to communicate with any device with a Bluetooth function, such as a mobile phone or laptop. It has two modes of operation, one is the data mode in which it can send and receive data from other Bluetooth devices, and the other is the AT Command mode where the default device settings can be changed. Schematic representation of the module in the figure is shown in the figure 2. (Fig. 2).

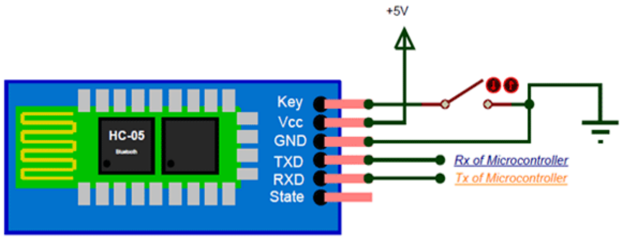


Fig. 2. Schematic representation of the Bluetooth module HC-05

E-ink or e-Paper indicates electronic use and mimics the appearance of normal ink on paper. It is useful when displaying text due to its high visibility and contrast, wide viewing angle and low power requirements. Once programmed, it remains unchanged until reprogrammed, even if there is no power. We used a 1.54 inch display. The advantage of this display is that under sunlight it has great visibility with a wide viewing angle of 180 degrees [6].

B. *Speech Recognition*

The speech recognition part of this solution is an Android application that communicates with the Google Cloud Speech API. The Google Cloud platform enables the development of applications that have the functionality to convert audio to text using neural network models.

This platform is a part of the system that recognizes spoken content in real time. Using the Google platform, a user's voice message is converted into textual content or transcript and displayed in the Android application. The user checks the authenticity of the converted content. Upon confirmation of accuracy, the transcript is forwarded and presented to another user. The implementation of frontend component is achieved in a few steps:

- Configuring Google Cloud API
- User Interface Design
- implementation of communication between Android application and Google Cloud API
- implementation of communication between Android application and Arduino UNO board

C. *Data Presentation*

The data presentation part of the system contains three components (Arduino Uno, Bluetooth module, E-ink display). This part of the system is based on the Arduino Uno board on which the HC-05 Bluetooth module is attached, which communicates with the Android phone. The presentation of the received content is done on the Waveshare e-Paper display connected to the Arduino board. The implementation of this part is divided into two parts based on the use of two modules:

- implementation of communication
- implementation of text printing on display

III. Description of Speech Recognition

A. *Configuring Google Cloud API*

Machine learning is part of the Google Cloud platform in creating applications that can hear, see and understand the world around them. In pretrained machine learning models, the Google Translate API and the Cloud Vision API are integrated into the Google Cloud Speech API.

By applying the already mentioned model of neural networks that converts an audio sample into a text format, it gives us the possibility of conversion in several ways. By recognizing sound in real time through a microphone through the application, sending an already recorded file from local memory or uploading to Google Storage.

This platform currently supports more than 125 languages [7]. The following steps were used to configure this platform:

- Speech API Approval
- Creating access credentials
- Creating a private key

B. *User Interface Design*

A simple design of the Android speech recognition application is described in the following two phases of work.

During the first phase, a screen is displayed waiting for the user to record a voice message.

Through the second phase, a screen is displayed after the user records a voice message the application will send the data to Google Cloud. After processing the data on the platform, the input voice is converted to text and displayed in TextView.

C. Implementation of communication between Android and Google Cloud API

As mentioned above, the API allows audio sample conversion in a number of ways. Realtime sound recognition was chosen in this study. By sending a request from the application, it is possible to get temporary and final recognition results in response. Provisional results are represented by current recognized patterns, while the final result is the best hit.

Calling the Streaming Speech API requires multiple requests, while configuration and audio are sent in a single request. The StreamingRecognizeRequest function imported within the com.google.cloud.speech package is used to forward platform requests [8].

Parts of the audio recognition are sent in sequential StreamRecognizeRequest messages. The first message must not contain audio content, while all subsequent StreamingRecognizeRequest messages must contain audio content. Audio bytes must be encoded as specified in RecognitionConfig. StreamingRecognizeConfig consists of the fields shown in the table (Table I).

TABLE I
DESCRIPTION OF STREAMINGRECOGNIZECONFIG PARAMETER FIELDS

Parameter fields	Description
config	contains information for audio, type RecognitionConfig
single_utterance	(default value is false) indicates whether the request automatically terminates if there is no more speech detection
interim_results	(default value false) indicates that this stream request should return temporary results that can be refined later

For StreamingRecognize requests, the sound must be sent at a speed that roughly corresponds to real time. Attempting to process content that exceeds the defined limits will result in an error.

A SpeechContext resource, in which one word counts as a phrase, contains a list of request-specific phrases which can also be provided with any request. The limit on phrases per request is a maximum of 5,000, 100 characters per phrase, and a maximum number of 100,000 characters [9]. A maximum of 900 requests per 60 seconds is possible, while each StreamingRecognize session is considered a single request even though it includes multiple StreamingRecognizeRequest sound frames in progress.

Through the RecognitionConfig, the parameters of the encoding type, the language code and the speed pattern in Hz type integer. Sample speed in Hz audio samples sent in all RecognitionAudio messages. Valid values are: 8000-48000. For best results, set the sampling frequency of the audio source to 16000 Hz. If this is not possible, the original sample rate of the audio source is used. This field is not required for FLAC and WAV audio files, but is required for all other audio formats. The Google Cloud platform does not support stereo audio files. LINEAR16 is used for AudioEncoding and Serbian is selected as the recognition language under LanguageCode, sr-RS is forwarded.

To respond, you need to create a StreamObserver array of StreamingRecognitionResponse type. The answer consists of the fields shown in the table below (Table II).

TABLE II
DESCRIPTION OF STREAMINGRECOGNITIONRESPONSE PARAMETER FIELDS

Parameter fields	Description
error	retuns Rpc status message
results	contains zero or one is_final = true (newly formed part), followed by zero or is_final = false (provisional result)
speech_event_type	type of speech event

StreamingRecognizeResponse is the only message returned to the client by StreamingRecognize. A series of zero or more StreamingRecognizeResponse messages is returned to the client. If there is no recognizable sound, and single_utterance is set to false, then no message is returned to the client. Based on the stability parameter, the one with the highest value is displayed by comparing the results (Table III).

TABLE III
DESCRIPTION OF SPEECHEVENTTYPE PARAMETER FIELDS

Parameter fields	Description
config	contains information for audio, type RecognitionConfig
single_utterance	(default value is false) indicates whether the request automatically terminates if there is no more speech detection

D. Implementing of communication between Android and Arduino UNO board

A transcript is obtained in response to the audio sample. Invoking the actionListener by clicking the "Confirm" button, the user makes a choice if the returned sample has a valid context, otherwise it is "Wrong". All successfully recognized transcripts are forwarded via Bluetooth connection to the Arduino UNO.

The connection for the HC-05 Bluetooth module used in operation was implemented. The required parameters for communication are its MAC address and UUID. BLUETOOTH and BLUETOOTH_ADMIN permissions need to be imported to AndroidManifest in order to enable Bluetooth communication.

During initialization, an attempt is made to establish communication with the Bluetooth module. First, it is checked whether the Bluetooth communication on the phone is switched on, if not, an Intent of the BluetoothAdapter.ACTION_REQUEST_ENABLE type is created, which starts the confirmation dialog.

Through the initBluetoothProcess() method, it establishes a connection with the module and creates a handler that receives information via ConnectedThread. This thread is in charge of communication with the module via BluetoothSocket. Each successfully sent message is answered with the value "success".

IV. Description of data presentation

The following steps show how to start sending a transcript of a recognized audio sample for printing on an e-Paper display:

1. By clicking the button, the user starts sending from the smartphone via Bluetooth communication
2. The Bluetooth module accepts the message and sends it to the Arduino
3. The Arduino processes the received message, which is printed on the Waveshare e-Paper display
4. The Arduino sends a message to the Bluetooth module, which returns this message to the smartphone

A. Implementation of communication

By importing the SoftwareSerial library, communication with the Bluetooth module is enabled.

First, it is checked whether there are incoming messages from the Bluetooth module, when receiving the message, byte by byte is read (reading the "char" type of each iteration). In order for messages to arrive successfully, it is necessary to pass 10 delay functions. Iteratively, byte by byte of the message is read and placed in the buffer. After the message is successfully placed, a flag is placed for the end and '\0' is added, which marks the end of the buffer. Finally, it is checked once more to see if there are any more incoming messages and all counters and buffer are set to zero.

B. Implementation of text printing on display

The epd.h library provided by Waveshare is used to present the text on the e-Paper display, to allow easy communication of the Arduino board with the screen. After initializing the screen, wake-up is activated using epd_wakeup function and the memory it uses during operation is defined. This display has the option of using its built-in NAND memory or external memory via SD card.

Receiving a message calls the printOnDisplay function. In order to clear the previous content from the display, the epd_clear function is used and the received message is printed with a call to the epd_disp_string function, which is forwarded with a text message, x and y positions. The screen content needs to be refreshed by calling epd_update function.

The screen used is low refresh rate and therefore it cannot be used to display animations, while in this research it proved to be effective.

V. System testing

The solution was implemented through four phases of testing divided into two parts. We used two languages, Serbian and English.

Each phase had its own peculiarities. In the first phase, random numbers from one to one hundred were spoken. From the second to the fourth phase, the complexity increases. In the second phase, two-syllable words were used, and with each phase the number of syllables was increased. Each of the phases used fifty words and the success of their detection was tested (Table IV).

TABLE IV
Table showing measured hits in four phases divided into two cycles

	I	II	III	IV
Serbian	41/50 82%	32/50 64%	26/50 52%	15/50 30%
English	47/50 94%	44/50 88%	39/50 78%	31/50 62%

This testing has shown that the success in recognizing the Serbian language decreases significantly with increasing number of phrases. Thus, we conclude that currently the better choice is the English language for a more authoritative functioning of our system.

VI. Conclusion

This paper presents one of the ways to use the Google Cloud Speech-to-Text API with the aim of speeding up and facilitating communication between two people.

Based on this research, for better functioning of the system, it is necessary to move the communication between people through the server. With this improvement, the use can be extended to more people without any distance restrictions. Adding the Google Translate API to the system, allows us to select the language for the presentation of recognized audio content.

References

[1] J L. Deng and X. Huang, "Challenges in adopting speech recognition," Commun. ACM, vol. 47, no. 1, pp. 69–75, 2004.

[2] Power, M.R. and D. Power, Everyone Here Speaks TXT, "Deaf People Using SMS in Australia and the Rest of the World," Journal of Deaf Studies and Deaf Education, 9(3): p. 333-343, 2004.

[3] Tucker, W., E. Blake, and M. Glaser, Building Bridges for Deaf Telephony in South Africa A Community-Centred Approach. 'Information Technology in Developing Countries', 13(2), 2003.

[4] M Stenman. "Automatic speech recognition An evaluation of Google Speech". UMEA UNIVERSITY. 2015.

[5] A. Glasser, K. Kushalnagar, and R. Kushalnagar, "Deaf, Hard of Hearing, and Hearing perspectives on using Automatic Speech

Recognition in Conversation," Rochester Institute of Technology., Rochester, NY 14623, 2019.

[6] Waveshare, "1.54 inch e-Paper Module", 2020. [Online]. Available: https://www.waveshare.com/wiki/1.54inch_e-Paper_Module#Arduino_UNO [Accessed 06.09.2020.]

[7] Google Cloud Speech-to-Text, "Language support", 2020. [Online]. Available: https://cloud.google.com/speech-to-text/docs/languages, [Accessed 06.09.2020.]

[8] Google Cloud Speech-to-Text, "API", 2020. [Online]. Available: https://cloud.google.com/speech-to-text/docs/streaming-recognize, [Accessed 06.09.2020.]

[9] Google Cloud Speech-to-Text, "Quotas & limits", 2020. [Online]. Available: https://cloud.google.com/text-to-speech/quotas, [Accessed 06.09.2020.]

The rising threat of hardware attacks: USB keyboard attack case study

P. D. Bojović, I. Bašičević, M. Pilipović, Ž. Bojović, M. Bojović

***Abstract:** The software aspect of computer security has been investigated more thoroughly in the research literature than the hardware aspect. Consequently, one of the aims of this paper is to raise public awareness regarding hardware-based attacks. During this project hardware that intercepts keyboard USB communication with a PC computer was developed. The hardware is based on Arduino modules and it can be used to execute several known attacks: keyboard logging, man-in-the-middle and scheduled BadUSB attacks. The research presented in this paper proves that by using commercial off-the-shelf hardware and rather simple software modules, efficient attacks that gain access to sensitive information and achieve control of the computer system can be realized.*

[1]*Index Terms*—computer security; keyboard; keylogger; men-in-the-middle; USB.

1. Introduction

The exponential growth of malicious software has led researchers to investigate this matter more thoroughly than hardware-based attacks. Third party hackers have skillfully used this situation to employ state-of-the-art commercial off-the-shelf components to target victims' devices, exploiting these vulnerabilities to gain access to sensitive information ranging from personal information to government-related data, critical infrastructure resources and sensitive enterprise data. One of the key components commonly present in the hardware-based attack assortment of assembly tools is a hardware-based keylogger (a highly specialized small electronic device, designed to intercept and record the keyboard input of a compromised machine) [1]. It permits attackers to steal large quantities of sensitive data stealthily, leading to malicious manipulation or data misuse. The attacker must be able to prepare access to the compromised system which can be done either during the design phases or in the utilization phase.

While they are more difficult to implement in respect to software keyloggers, hardware keyloggers can be more flexible and advantageous for the attacker as they do not use any computer resources, cannot be detected by antivirus software and are OS-independent. Some keyboards are even designed with built-in keylogger functionality [2]. There are also wireless keylogger sniffers that can intercept and decrypt data being transferred over the air between a wireless keyboard and its receiver. Of particular interest are attacks on bluetooth communication, where the implementation of MITM attacks on Bluetooth keyboards can provide access to all keystrokes typed on the keyboard [8].

Interest in keyboard attacks imposes the need to explore possibilities for different access to keystroke information. In this sense, it is possible to apply the tone recognition method of each key on the keyboard using a combination of standard machine learning and speech recognition techniques. It is also possible to analyze the vibrations generated by the push of a button. The information on the keys pressed can be read using the accelerometer or gyroscope.

Keyboards have become a common focal point for attack by cyber-thieves. The pervasive nature of keyboards has led to their commoditization by manufacturers and consumers. Unfortunately, history has shown that most consumers are not willing to pay extra for cheap hardware commodities, regardless of whether they embrace security enhancements. These choices are related to expected costs and benefits. Consequently, manufacturers are not motivated to improve the keyboard and make it more secure. As a matter of fact, most keyboard manufacturers despite knowing about the vulnerabilities of their products refuse to accept liability within the context of keyboard logging attacks and explicitly state that they are not liable in their warranties [3]. Due to a lack of awareness regarding hardware keylogger issues, companies selling products to counter the threat of these attacks and the economies of scale in their production, distribution and marketing receive far less attention than they deserve.

In the realm of information security risks, hardware keyloggers continue to pose an important problem. A deeper understanding of these silent and stealthy malicious devices and their impact on user privacy and security can lead to viable techniques for both protection and detection.

The aim of this research is to point out the possibility of a relatively easy and effective attack on USB keyboard communication. For that reason, we analyzed our custom-designed embedded system to demonstrate and evaluate the effectiveness of several known attacks. The hardware is based on Arduino modules. The procedure that performs the attack has also been written in script language. The research presented in this paper proves that by using commercial off-the-shelf hardware components and rather simple software modules, efficient attacks that gain access to sensitive information and achieve control of the computer system can be realized. Thus, one of the aims of this paper is to raise public awareness regarding keylogger intrusions. More significant research in this area should be conducted by both

[1] This research was financially supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia through Projects No. III 45003, III 44003 and III 44009-2.

P. D. Bojović aff. The School of Computing Union University Belgrade, 6/6 Knez Mihailova, Belgrade, Serbia

I. Bašičević, M. Pilipović and Ž. Bojović aff. Faculty of Technical Sciences University of Novi Sad, 6 Trg Dositeja Obradovića, Novi Sad, Serbia

M. Bojović is independent scientist

Corresponding email petar.bojovic@paxy.in.rs

industry and academia.

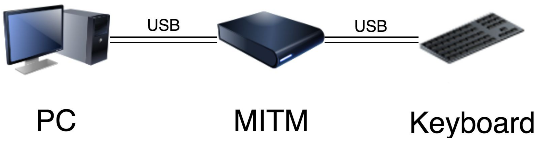


Figure 1 - USB MITM setup

The following sections are organized as follows. In Section 2 the related work is presented. Various types of investigated attacks are summarized in Section 3. A USB man-in-the-middle (MITM) [Figure 1] proof-of-concept attack, in which a set of hardware components were used to intercept a USB connection with the PC and therefore initiate an attack is presented in Section 4. Section 5 then describes the case study scenario. Section 6 comprises a discussion of the achieved results and finally, in section 7 the conclusions of the study and some future work are described.

2. Related work

Over the years, researchers have developed various protocols at the software level for the purpose of computer security. However, the attention of the researchers is in most cases occupied by protecting protocols for input devices. With the development of semiconductor processes and the emergence of open-source hardware there has been a rise in the manufacture of low-cost, high-performance embedded devices [4].

A previous paper [5] described how to detect software keyloggers that are capable of recording every keystroke made by users in a log file. It can record information such as user ID, password, instant messages or emails [6]. For this reason, the authors proposed a solution based on using signature and heuristic analysis to fight against keylogging.

With the development of semiconductor processes and the emergence of open-source hardware, there has been a rise in low-cost, high-performance embedded hardware. Researchers analyzed existing studies that used custom security inspection tools and developed IRON-HID, a framework tool with superior and more accurate functions than the existing tools [4]. This tool is programmed with the Arduino framework and is designed to be attached to the existing hardware. It offers keyboard emulation functions that monitor and transmit keyboard entries and a function that simulates the mass-storage device to install malware.

Today, personal computers have become the main weak-points for hacker penetration testing. Another study testing PC machines was conducted using a USB Rubber Ducky device which behaves like a keyboard and obtains clear text login ID and passwords from a computer. This problem was analyzed in a previous paper [6] in which the details of a penetration into a Windows machine with antivirus software via a USB Rubber Ducky device and scripting are described.

Technology improvement requires the development of new computer security methods, besides detecting keyloggers. A previous study [7] explained how unprivileged software running on a PC host can replace the firmware on an off-the-shelf gaming mouse. It also shows how compromised firmware on the mouse can in turn reinfect the PC host. The attacker could deliver to users a seemingly legitimate mouse firmware update via a MITM attack, by posing as the mouse manufacturing company. The authors considered two separate attack models: one in which the mouse is connected to a machine with Internet access and another in which the mouse is connected to an air-gapped machine (a machine isolated from the Internet). There are several other options for triggering a firmware update including timers based on user inactivity, communication with an adversary-controlled program on the target and fingerprinting of the target.

USB has become the major connection port of modern computers and it provides programmability as a convenient way for hardware and OS vendors to develop their products and related firmware. However, it leads to high risk by opening the door for a potential vulnerability because of its simple plug-n-play design. In one study [8], the researchers identified the high risk associated with BadUSB as a device that emulates the keyboard or network card. One such device can register as a flash drive and a keyboard at same time, which allows an invasion of malicious software. Since the host is not able to scan the firmware, antivirus software cannot identify this threat [9].

The most important part of BadUSB is the MITM attack which allows access to all the confidential data from the computer user. A MITM attack on a connection between a Bluetooth keyboard and a computer has been presented [8], which provides the same functionality as a keylogger on a Bluetooth keyboard and which will appear to the user to be nothing more than a device malfunction.

Researchers have developed a solution called “Good USB” as a mitigation architecture for the Linux USB stack. They observe that the root cause of such attacks is that the Linux USB stack exposes a set of unrestricted device privileges and note that the most reliable information about a device’s capabilities comes from the end user’s expectation of the device’s functionality. It defends against BadUSB attacks by enforcing permissions based on user expectations of device functionality [9]. Current counter-measures either require special USB devices or ask the user to decide if the device can be used. Another implementation for better security involves forcing the user to interact with the device physically, to ensure that a real human-interface device is attached. The focus is on the authorization of the mouse and keyboard while inhibiting the use of other human interface devices (HIDs) such as joysticks. However, this approach has an obvious limitation: it cannot prevent a malicious HID from actually allowing the user to enter the challenge code [10].

3. Attacking method

In this paper we present research covering the development of an undetectable malicious keyboard. The goal of this research is to demonstrate the ability of malicious keyboards to initiate attacks:

MITM attacks, key-logger functionality, BadUSB attacks and scheduled attacks.

3.1 Man-in-the-middle attack

The existing technological solutions use standard USB to establish communication between the PC and the keyboard controller. This communication is a simple serial signal exchange. Since it does not use authentication of the other party, it is possible to inject a module between the two sides of communication. A hardware USB Master and Slave is needed to successfully establish USB communication between intended parties while obtaining data about the traffic passing between them. The USB Master (USB Host controller) is part of the MITM that will present itself as the PC to the keyboard. The USB Slave (USB emulator) is used by the MITM to present itself as a keyboard to the PC. The main functionality of the MITM attack is to transmit the data received from the keyboard (USB Host controller) to the PC (USB emulator) so that keyboard functionality can remain unchanged.

3.2 Key-logger functionality

Having implemented a MITM attack, additional keylogging functionality becomes trivial. If the MITM device has access to the SD-card or any other form of non-volatile memory (such as a flash drive), it can transparently store any data coming from the keyboard. Any conventional way of handling a hardware keylogger, such as anti-viruses or an Application Level Firewall, would be unsuccessful. The reason is simple, all interception and data logging features are implemented in the hardware inside the keyboard (from the PC perspective) and the system software can only see key actions (pressed, released etc.).

3.3 BadUSB attack

This attack allows an attacker to activate instructions prepared in advance that the keyboard emulates as if the keys were manually input. By scripting instructions inside a text file on the SD-card the attacker could send the instruction to the victim PC to for example, create a new blank file, enter some text into a program such as VBS, BASH, PowerShell, etc., then save this file and execute it. Theoretically, it is possible even to retype an entire binary malware file and execute it [11]. BadUSB as an attack is mostly undetectable by current anti-malware software as it presents itself as regular keyboard input. Most of the malware files created using BadUSB scripts are undetectable as they are retyped into file, not copied from some other source. BadUSB could be effectively used as a loader malware attack that would simply create a small script that downloads from the network or Internet some complex script or binary for a more sophisticated attack. A BadUSB attack acts as the keyboard, so it works in the foreground of the user interface. If there is a user present at the moment of the attack, the keyboard misbehavior will be visible. However, the speed of key execution could be so fast that the user has no time to react and disconnect the keyboard.

3.4 Scheduled attack

To avoid the detection of a BadUSB attack, we propose that the attack have a time scheduled trigger. If the attack is initiated as soon as the keyboard is attached to the PC, it would be too obvious that it contains a malicious component. Instead, the attack should be delayed or even better, scheduled for a specific time when the attack has the best chance of success.

MITM hardware should be deeply embedded into the keyboard itself to avoid detection. The SD-card slot on the other hand should be relatively easily available so that the attacker can access the logged data and re-upload the BadUSB script instruction. It is probable that the best positions for SD-card access are under large keys such as the Enter or Space buttons. Getting access to the keyboard and SD-card is part of social engineering and is not included in this research. As an alternative to switching SD-cards, the attacker could switch the entire keyboard with one that is exactly (or close to) the same.

4. Proof of concept

As a proof of concept, we have developed a prototype that includes the set of hardware components needed to intercept a USB connection with a PC and therefore initiate a MITM attack [Figure 2].

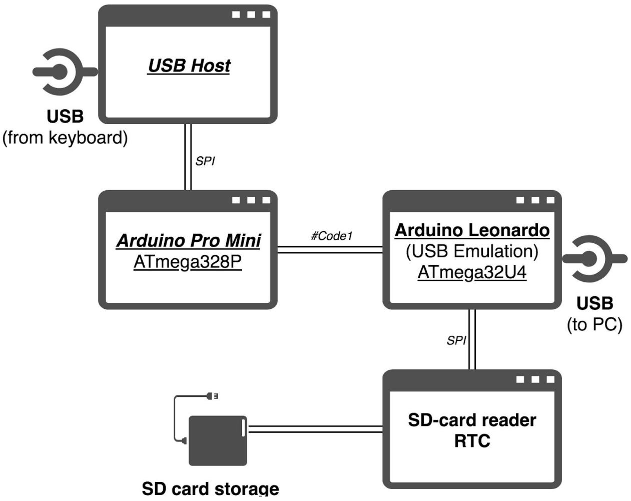


Figure 2 - The module scheme for hardware MITM concept

4.1 Hardware

The USB Host module is a module that provides HID connection to Arduino developer boards. Its function is to communicate with the USB keyboard and forward signals from it to the Arduino board. Our implementation used a USB Host Mini shield that communicates with Arduino over an SPI interface.

The Arduino Pro Mini module is a dedicated micro-controller tasked with receiving and processing the signals received by the USB Host controller and keyboard. The main role of this module is to process and retransmit the ASCII codes of pressed and released keyboard buttons. The board is based on the ATmega328P microcontroller. The USB Host Mini shield is designed for the Arduino Pro Mini so hardware connections are established by simply connecting their pins.

The SD-Card/RTC module is a two-in-one module with peripherals. The SD-Card communicates with the Leonardo board over the SPI interface to allow data reading/writing on the controller. The real-time clock (RTC) is a battery powered module that is able to keep the clock persistent even when device is without power. The RTC establishes communication with the controller over an I2C interface.

The Arduino Leonardo developer board module was used for the implementation of the functionality described in this paper. It is based on the ATmega32U4 microcontroller. The module is able to emulate any kind of USB HID device. The Leonardo board is connected with the Pro Mini over the UART (serial) port so that it can receive the ASCII codes once keys are pressed or released. The Arduino Leonardo board variant we used has a USB type A male connection in the PCB board, so it can be easily plugged into a PC or connected to a USB extension cord.

4.1.1 Power issues

The Arduino Pro Mini uses 3.3V to power the ATmega328P controller. Such a voltage cannot be used natively from a USB connection as USB devices are standardized to 5V. A DC-DC voltage regulator is required to prepare a stable 3.3V power supply using a 5V source. Fortunately, the variant of the Leonardo board we used has exactly such a regulator (AMS1117).

The USB Host module, by data-sheet, should have support for both 3.3V and 5V. That was not the case with the board selected for this study. With 5V on the USB Host we could not establish an SPI connection between the USB Host and Pro Mini. Also, with a 3.3V supply there have been certain issues as the USB keyboard connection was unstable (the USB standard requires 5V). Connection with the USB Host over SPI could be established and keys from keyboard received, but every 10 seconds the USB Host would reboot and lose connection. The solution was a small hack of the USB Host module. The bus connection to the 5V (VCC) pin on the USB connector was broken and instead directly connected to the 5V power supply from the Leonardo board. With this hack, the USB Host module was supplied with 3.3V for SPI communication with the Pro Mini and the keyboard received a standardized 5V power supply. This hack solved the stability issue.

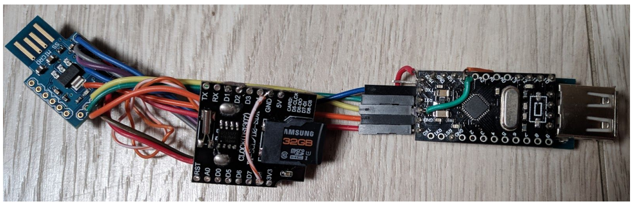


Figure 3 - Working prototype before keyboard integration

4.2 Software

The concept was programmed in Arduino as it contains all the necessary libraries and examples to make our task as simple as possible. The idea was to implement all proposed attacking techniques inside the controllers. Unfortunately, both the Pro Mini (ATmega328P) and Leonardo (ATmega32U4) boards have limited programming resources. Both have only 32KB of programmable flash memory and part of it is reserved by the bootloader. In starting this project we found that we could not implement all the described functionality in such a limited code space. As a consequence, the code had to be split into two modules presented in Figure 4. By design, one module handled communication with the keyboard while the other module performed all the other tasks.

4.2.1 Keyboard communicator

The Arduino project is designed for the Arduino Pro Mini board that communicates over the SPI port with the USB Host Mini shield. The USB Host communication library for Arduino was developed by GitHub user Felis [12]. It establishes an SPI connection with the USB Host module and prepares listeners for key press events. In the case of events the declared method is called. As the intention in this case was for it simply to handle press and release button events, others have been ignored such as the state changes of control keys, etc.

The idea was to receive a key code on events and redirect it to a serial UART communication port. But there was a problem; the received codes were not ASCII coded characters. Instead, they were OEM table coded. The library provides a method for translating OEM codes to ASCII, but that fails if the code refers to a group of control keys such as Alt, Ctrl, Shift, arrow keys, etc. We had to create a method that would translate even control buttons to the ASCII code understandable by Leonardo so that it could use them for emulation. For most of the keyboard control buttons the OEM code was simply replaced with adequate ASCII code. For control buttons that are often used in combination with others (e.g., Alt, Ctrl, Shift and GUI (Windows)) we added a flag so it was transmitted with every pressed button. It was also designed to flag if a press or release event was transmitted (+ if pressed, - if released).

Finally, the Pro Mini communicated with the USB keyboard and sent ASCII code in event cases as seen in Code 1.

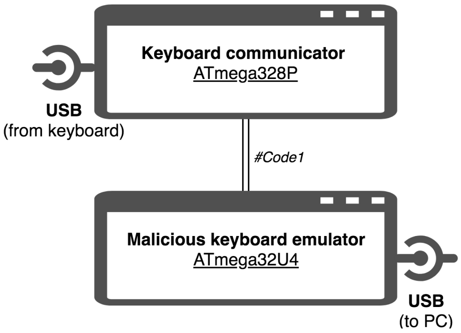


Figure 4 - Software module scheme

```
61C+   witch means char ‘a’ (hex 0x61) is pressed (+) together
       with “Ctrl” button ( C )
 B2+   witch means space (hex 0xB2) is pressed (+)
 B2-   witch means space is released (-)
```

Code 1 - Example and description of protocol codes

4.2.2 Malicious keyboard emulator

The main part of this concept was programmed into an Arduino Leonardo variant board that uses an ATmega32U4 microcontroller. It initiated a serial (UART) connection with the Pro Mini in order to receive the ASCII codes of pressed keyboard buttons. The main component of this project was the keyboard emulation library integrated into the Arduino SDK. It sets the ATmega32U4 to a keyboard emulation mode so that the PC computer recognizes the device as a regular HID keyboard. With such settings the ATmega32U4 acts as a regular keyboard, so no additional drivers are required to be installed on the PC.

In order to accomplish a scheduled BadUSB attack, we needed to keep accurate time in the micro-controller. This is not a trivial task as the micro-controller cannot keep time after losing its power. We therefore needed to add RTC hardware that would be able to keep time even when the “keyboard” was disconnected. At the beginning of controller initiation, it needs to resynchronize its internal clock with the RTC. This operation allowed us to schedule the future attacks.

Communication with the SD-card controller using the SPI interface was also established and checked. The SD-card contains some necessary instructions for the attack, in addition to stored data received by the Pro Mini. After successful initiation of the SD-card, we accessed the “time.txt” file that contained the date and time for the scheduled attack. That time was stored in local memory so that it could be compared later with the system time.

After the initiation part is complete, the controller enters the loop stage. Once in the loop stage, it checks if there is data available on serial communication with the Pro Mini. It also verifies whether it is time to initiate the BadUSB attack.

Once a key is pressed or released, Code 1 is received over UART and the string is parsed. The first two characters are ASCII code in hex format, so they are converted to ASCII byte. The rest of the string is evaluated as flags for key presses, releases or special control keys. If a “key pressed” flag is detected, the ASCII byte is sent over the controller to the PC which it interprets as a keyboard key press. If the “key released” flag is detected, the release key is sent to the PC. If any of the control key flags are detected, that control key code is sent before the ASCII byte.

As soon as the key is sent to the PC, it is recorded in an internal string buffer for a key-logging feature. The buffer is able to store up to 1000 characters before it writes them to the SD-card. If the buffer is filled or one minute passes since any button is touched, the buffer flush is initiated. At that moment, the controller accesses the write-append privilege to “datalog.txt” file and appends the content of the stored buffer. After, it closes the “datalog.txt” file and clears the buffer. Buffering is required in order to optimize the time required for SD-card write operations.

The controller loop checks whether the time set to the local value matches the current system (synchronized with RTC) time and if so it initiates the BadUSB attack. In that moment, the controller stops its usual behavior (retransmitting received ASCII code to the PC and recording) and enters the attack procedure.

It accesses the SD-card’s “script.txt” file that contains the DuckyScript-based [Code 2] instructions for the attack. Such a script is a list of instructions that tells the controller what is to be sent to the PC.

An example of such a script has been previously presented [13]. The example was created for a Windows OS and demonstrates how a VisualBasic Script (VBS) file that can load any malware can be created. The presented code performs the following operations:

1. Starting “Notepad”.
2. Accessing the internal file named “run.vbs”.
3. Re-typing its content to the opened empty Notepad workspace.
4. Saving the content to a file at the temporal location.
5. Running this script using Windows RUN.

Delays are added to slow down the controller’s instruction as some time is required before the PC is ready for the next instruction. REM are comments, STRING is the string to be sent to the PC and lines such as “GUI r” – represent instructions to send the GUI key (Windows key) then press the “r” key.

Code 2 is an example of a DuckyScript that creates a malicious file containing the VBS “run.vbs” [Code 3] and runs it on the system. The VBS simply displays a warning message and deletes itself.

The controller runs the BadUSB attack for each line in the “script.txt” file. Once it finishes the interpretation of all lines, it switches back to its normal state – retransmitting received ASCII code to the PC and recording the keys to a file.

```
REM Wait for the system to get all set up
DELAY 3000

REM Start Run
GUI r
DELAY 500
REM Enter to Notepad
STRING notepad
ENTER
DELAY 500

REM Type content of VBS file
PAYLOAD run.vbs
REM Save file
CTRL s
DELAY 500
STRING %TEMP%\run.vbs
ENTER
DELAY 500
ALT F4
```

```
REM Run VBS file
DELAY 500
GUI r
DELAY 500
STRING %TEMP%\run.vbs
DELAY 50
ENTER
```

Code 2 - DuckyScript example that create malicious VB script and execute

```
x=msgbox("Virus is just started ! (Paxy)" ,0, "You are hacked !")
Set objFSO = CreateObject("Scripting.FileSystemObject")
strScript = Wscript.ScriptFullName
objFS
O.DeleteFile(strScript)
For i = 1 to 5
Wscript.Sleep 1000
Next
```

Code 3 - Example of malicious VB script

5. Testing scenario

In order to test the developed malicious keyboard attacking tool, we conducted an experiment with a member of our research team. The team member had knowledge of the prototype [Figure 3], but was not aware that he would be the subject of a testing scenario that is very close to a real-life pen-testing scenario. The idea was to execute a multi-phase attack with the final goal of gaining remote access to the user's PC.

The first phase was cloning the target keyboard with a malicious one. Luckily, the target keyboard was a common workstation keyboard that could be easily obtained from off-the-shelf computer stores. An identical keyboard was bought and space made for prototype hardware integration. The USB cable was cut and routed over the prototype which decreased the cable length by a small amount. Under the Enter key the SD-card slot was opened and the entire prototype sealed with hot glue [Figure 5].

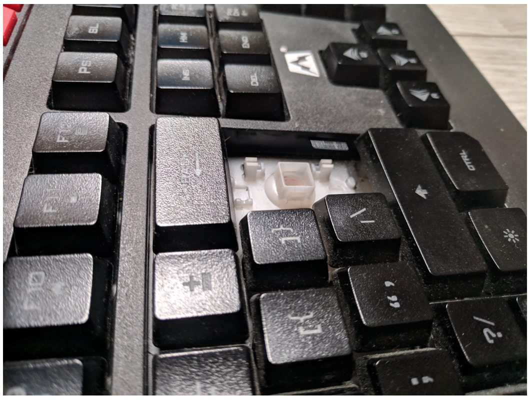


Figure 5 - SD-card slot hidden under the Enter key

The second phase was to construct a scenario for switching the keyboard that would pass unnoticed. This part included a trust exploiting technique as part of social engineering. We took the opportunity to switch the keyboards during a moment of the researcher's brief absence.

Keylogging capabilities are the default prototype feature as long as the SD-card is present. The malicious keyboard was placed and planned to record keystrokes for one week.

The third phase was to find a way to access the keyboard in order to collect and swap the SD-card. The collected keystrokes data allow the development of the final phase of the attack. Again using some social engineering techniques, the target was convinced to let his PC be used for a moment. It was enough to gain access to the keyboard, remove the Enter key and swap SD-cards without raising suspicion.

The fourth phase was to analyze the keystrokes and find a string indicative of a PC login password. This data can then be used to create a DuckyScript to emulate user login keystrokes, gain access to the system and initiate some pre-installed software. The researcher was known to be a highly professional user who employs remote access tools such as VNC for PC remote control. Our plan was to initiate a stealth VNC server and remotely connect it to the VNC viewer listener. That way a reverse VNC connection to the listening VNC viewer can be created.

With completion of the developed script and testing in local computers, we were ready to finalize the attack by scheduling script execution for the time with the least chance of detection.

The fifth phase included another social engineering attack in order to gain access to the keyboard once again. This time, the malicious keyboard was loaded with the SD-card containing the script content for execution of the final attack at a specific time.

To ensure that the target computer was powered on and the user was away from his computer, we caused a distraction. The attack was scheduled to be executed at dinner time. At this moment, we made a call to the researcher using a landline phone and asked him some questions about his daily job. A few moments later, a fully controllable remote desktop connection was established from his computer.

After talking with the researcher about this scenario, he found only the social engineering methods we used to be strange. He noted that we asked him if we could use his computer more often than before. It was also highly unusual to call someone over landline in the era of personal cell phones. However, he did not notice any changes with regard to the keyboard.

6. Discussion

Hardware-based attacks should receive greater attention as they are much easier to disguise. In addition, they are rather difficult to defend against with current state of technology. The majority of existing hardware keylogger or BadUSB devices are in the form of two-port adapters that are intended to be placed between the PC port and USB keyboard. This makes them potentially detectable as a usual place to find them is close to a PC port. Unfortunately, based on experience we can conclude a noticeable lack of practice in verifying the keyboard connection with the PC prior to the use of a public PC for accessing private accounts. Even within security experts, trust in hardware keyboards is high.

Generally, computer keyboards are made with different shapes, sizes, number of keys and are even designed for special purposes. Whatever the type of keyboard, the function remains the same – enabling the input of characters using fingers. Even as keyboards vary in size, button size is generally the same so they can be finger friendly (close to 19 mm [14]). Size variation is usually caused by the number of buttons (usually 84 - 104 keys) or design. If the separation between some function keys is considered it can certainly be said that keyboards must be large to be comfortable to work with. They are not complex electronic devices. With simple electronics and a few flexible sheets of conductive buses they accomplish their basic functionality. As such, the primary requirement with respect to size is user comfort.

Consequently, with their large size and simple electronics logic, there is enough space to integrate custom attacking electronics inside the keyboard itself. There is no necessity for an external or unusual device that is visible and hardware attacks become significantly less detectable.

7. Conclusion

Different attacks on computer systems and their components, whether through hardware or software, are constantly receiving the attention of the scientific and professional community. The goal of this paper is to raise awareness within the community with respect to potential threats coming from hardware attacks. We analyzed several types of keyboard hardware attacks (i.e., keylogger, MITM and BadUSB attacks). These attacks were tested on the Arduino hardware platform, which indicates that the implementation of these attacks is rather simple and has a huge impact on computer security and security in general. The tests results clearly point to the fact that a hardware keylogger disguised within a keyboard is extremely difficult to detect due to the high diversity of keyboard sizes and shapes.

Such a simplification in the implementation of the attack hardware indicates that is necessary to pay more attention to this type of attack and expend significant effort in finding more effective protection models. For these reasons, our future research will be related to the development of countermeasures. One idea is to develop and implement security mechanisms such as a digital signature and digital encryption between the computer OS and peripherals.

8. References

[1] Sagiroglu, Seref & Canbek, Gürol. (2009). Keyloggers Increasing Threats to Computer Security and Privacy. Technology and Society Magazine, IEEE. 28. 10 - 17. 10.1109/MTS.2009.934159.

[2] Caloyannides, Michael A. Privacy protection and computer forensics. Artech House, 2004.

[3] Hewlett - Packard, *Hewlett-Packard Development Company, L.P.* HP Worldwide Limited Warranty and Technical Support, 2015

[4] Han, Seunghun, Wook Shin, Junghwan Kang, Jun-Hyeok Park, HyoungChun Kim, Eungki Park, and Jae-Cheol Ryou. "IRON-HID: Create your own bad USB."

[5] D.B.V Jagannadham, D.Ajit Varma, “CYBERCRIME SECURITY IMPLEMENTATION USING HARDWARE BASED ANTI–KEYLOGGER” , Recent Innovations in Science, Technology and Engineering, https://www.ijarse.com/images/fullpdf/1513755373_122.pdf [Online; accessed August 2020]

[6] Cannoles, Benjamin, and Ahmad Ghafarian. "Hacking Experiment Using USB Rubber Ducky Scripting." Journal of Systemics 15, no. 2 (2017): 66-71.

[7] Cuthbert, Torrey & Gontarek, Andrew & Jensen, Elizabeth & Robbins, Peter. (2019). A Bluetooth Keyboard Attack.

[8] Almazaydeh, Laiali & Zhang, Jun & Wu, Peiqiao & Wei, Ruoqi & Cheng, Yisheng & Elleithy, Khaled. (2017). Bad USB MITM: A Network Attack Based on Physical Access and Its Practical Security Solutions. Computer and Information Science. 11. 1. 10.5539/cis.v11n1p1.

[9] Tian, Dave Jing, Adam Bates, and Kevin Butler. "Defending against malicious USB firmware with GoodUSB." In Proceedings of the 31st Annual Computer Security Applications Conference, pp. 261-270. ACM, 2015.

[10] Griscioli, Federico, Maurizio Pizzonia, and Marco Sacchetti. "USBCheckIn: Preventing BadUSB attacks by forcing human-device interaction." In 2016 14th Annual Conference on Privacy, Security and Trust (PST), pp. 493-496. IEEE, 2016.

[11] P. D. Bojovic, Cjmcu Badusb re-create binary file using ducky-script, base64, certutil, https://youtu.be/8sIlUc57eXw [Online; accessed August 2020]

[12] Github user *Felis*, USB Host Library for Arduino, https://github.com/felis/USB_Host_Shield_2.0 [Online; accessed August 2020]

[13] Github user hak5darren, Ducky Script, https://github.com/hak5darren/USB-Rubber-Ducky/wiki/Duckyscript [Online; accessed August 2020]

[14] Computer Keyboards, http://www.quadibloc.com/comp/kybint.htm [Online; accessed August 2020]

SEO in Serbian websites

Jana Knezevic, Jovana Panovic, Nikola Savanovic, Miodrag Zivkovic

***Abstract*—Search Engine Optimization, well known as SEO, strives to improve sites' visibility on search engines. It is some sort of guidance on how to become the first result in search results. In its practice is to improve/make the most of quality and quantity of website traffic. It is important to emphasize that SEO is all about understanding what people are interested in and what they seek to find. Because Global network and optimization techniques keep changing dynamically and the new ones are appearing, it is very important to keep up with the latest SEO news. We found some of the most important factors that might help improving the position in organic search results. With this in mind, we have aimed to discover the most common mistakes when it comes to our (Serbian) sites. This study will be beneficial to website developers, webmasters and any other person who participate in creating our websites. While doing this research, not to mention that we were testing and validating all results. The last part of our article concludes the whole story of our SEO research and gives further recommendations.**

***Index Terms*—search engine optimization, search engines, SEO, page rank, website**

I. Introduction

Search engine optimization is a series of modifications and techniques that should lead a site to be on the first page of search engines. Which one will be used, depends on the target of SEO, the type of website and website owner's main goals. SEO is more about improvements of unpaid results, known as "natural" or "organic" results and less about the purchase of paid placement. SEO aspires to help search engines to understand and present content, but it is more focused on users experience. Since Google started in 1998, their mission has always been to organize the world's information and make it universally accessible and useful. Under those circumstances, it is equally important to be focused on both Google's algorithm and users needs. There is no secret formula to be on the top of the first page and, of course, no one outside the Google knows the exact algorithm. Some studies showed that 95% of people never go further than the first page, nearly 75% of searchers start their searches on Google and the first five results on Google get 67% of all clicks [1].

With all this in mind, this study's purpose is to explore and disclose some of the top ranking factors and besides that, to demonstrate some of the most frequent mistakes with a tendency of making improvements.

II. Search Engine Optimization

As it is mentioned earlier, search Engine Optimization (SEO) is a long-term strategy, which should lead to a higher ranking on search results page and, subsequently, bring higher traffic to a website.

The working process of search engine optimization has three steps. Firstly, spiders crawl the web as they visit web pages, then they add optimized pages to their index and sort them so Google can show the best results based on what people search for.

The most popular search engines, Google, Bing and Yahoo!, use crawlers to help them find pages for their algorithmic search results. Each of these search engines are the answers machines and their main goals are to find, understand and sort the information only because of the mission we have mentioned before.

Equally important is search engine results pages, referred to as "SERPs". These are web pages that users see when they look up for something while using a search engine.

Not to mention there are large quantities of pages on the web, but none of them are more important to all types of profession than SERPs. Also, because the SEO techniques are user oriented, it is understandable why the answer boxes have become very popular lately.

There are two types of SEO. On-page SEO directs to the things on your site that you can optimize it is also under direct control of a developer. Main elements of On-page SEO are: title, content, URL structure, headings. Off page SEO is influenced by some external factors, such as readers, other developers, publishers and visitors. These factors can boost traffic of a site and its main elements are: blog, forums, social networks and so on... Off-page SEO makes reference to some actions that are not taken on your site and it is about creating good site reputation through high quality content.

SEO techniques can be sorted into two main categories [2].

- Techniques that search engine companies recommend, also known as white hat. White hat advice is generally summed up as creating content for users, not for search engines, and then making that content easily accessible to the online "spider" algorithms.
- Techniques that search engine companies does not approve, known as black hat. Black hat SEO tries to improve rankings in ways that are not approved by the

Jana Knezevic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: jana.knezevic.18@singimail.rs).

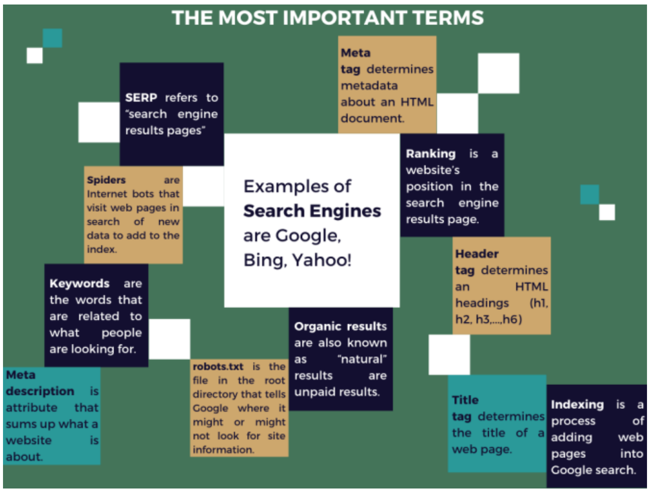
Jovana Panovic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: jovana.panovic.18@singimail.rs).

Nikola Savanovic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: nsavanovic@singidunum.ac.rs).

Miodrag Zivkovic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: mzivkovic@singidunum.ac.rs).

search engines.

The third category is grey hat SEO and it is totally based on improving search engine rankings, as shown in Fig.1.



THE MOST IMPORTANT TERMS

SERP refers to "search engine results pages"

Meta tag determines metadata about an HTML document.

Spiders are Internet bots that visit web pages in search of new data to add to the index.

Examples of **Search Engines** are Google, Bing, Yahoo!

Ranking is a website's position in the search engine results page.

Keywords are the words that are related to what people are looking for.

Header tag determines an HTML headings (h1, h2, h3,...,h6)

Organic results are also known as "natural" results are unpaid results.

Meta description is attribute that sums up what a website is about.

robots.txt is the file in the root directory that tells Google where it might or might not look for site information.

Title tag determines the title of a web page.

Indexing is a process of adding web pages into Google search.

Fig. 1. The most important terms in SEO

III. TOP RANKING FACTORS

Between the months of July and August we have been working on this project. The main purpose was to validate top 30 Serbian sites. The process of validation was simple. We have used the most recommended CSS and HTML validator. We have copied URL addresses in these validators, and by clicking on the option validate, errors would appear. Also, because of the privacy terms, we would not name any specific site. There were many errors, but we have summed up the most repeated ones.

First thing to remember and very important to be aware of, as we have mentioned earlier, is that there is no secret formula to be on the top of the first page. Not to mention that SEO ranking factors keep changing during the time and with that in mind, it is necessary to maintain and improve the knowledge all over again.

However, here are some authors [3] recommendations on how to boost traffic through site that we have explored and described.

A. A secure and accessible website

A secure and accessible website.

Google needs to see site's URL and look at the content in order to understand what the page is about. For this, one will need:

- A website created with a well-coded website builder
- A robots.txt file
- A sitemap that lists all your pages

B. Page Speed

Google wants to improve experience of users. If your site does not load fast on mobile devices, then it could be penalized. So, in that case, use Google's mobile testing tool to see how your site stacks up. One can start using Google Search Console in order to update site's performance, including speed.

C. Mobile friendliness

Nowadays more and more people uses mobile devices rather than desktops. If one's site is not mobile-optimized, the risk of getting needlessly under-ranked is high.

D. Domain Age, URL and Authority

Very few younger sites achieve ranking such as some older sites do (three years old and more)

Authority. It is a combination of great content and off-page SEO signals like inbound links and social shares.

E. Optimized content

LSI keywords - latent semantic indexing keywords. Word association, for example when you type "liverpool" there will be results for football club - Liverpool F.C. instead of the city in UK.

You should understand users' search intent in order to optimize well your content.

Other ways to optimize well:

- Content over 2k words
- Answer boxes
- Using videos

F. Author's suggestions

Here are some suggestions when it comes to importance of getting the code right:

- Add keyword phrases in page titles, which is where Google first look to determine which content is relevant to which search. One of the best tool for finding the relevant keywords is platform called *ahrefs*.
- Use header tags to show content hierarchy starting with your title at h1 and then use *h2* or *h3* for subheads
- Create a meta description that both entices readers and includes your keyword phrases
- Keep those meta descriptions short and catchy at around 160 characters
- Use keyword phrases in image alt tags to show how those images are relevant to the main content
- Use schema markup to tell Google what kind of content you are producing.

G. User experience

Man-made intelligence is used to better rank web pages and it includes:

- Click-through rate (CTR) - the number of users who click on your site after Google shows them their search result
- Bounce rate - it is almost the same as the previous one, but the difference is that people go back quickly to their search results
- Dwell time - it is the time visitors stay on your website.

H. Links

Off-page SEO often refers to Link Building. That is the process of getting other websites to link to your webpages units, as shown in Fig. 2.

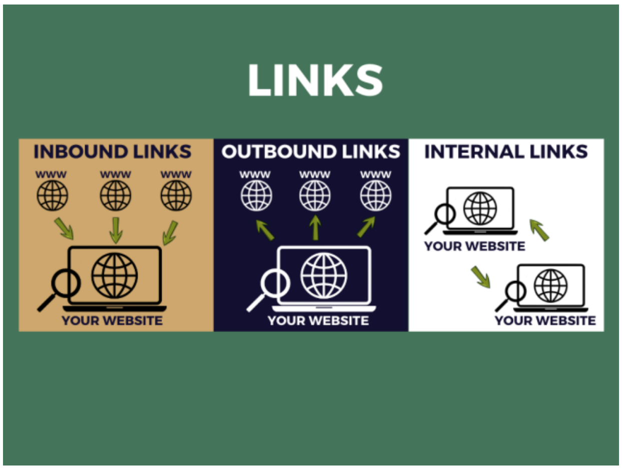


Fig. 2. Process of Link Buildings

I. Social media and signals

Of course, it is important for your content to be shared between people; it means it is precious and worthy. Therefore, the best advice is to start a blog. That can and will help your customers to solve problems. There we have some tools to help us with that, such as Smash Balloon and OptinMonster.

Social media, such as Instagram or Youtube, are a huge part of Global network, especially when it comes to search results. Whenever one is searching for their favorite movie star or hotel, the very first results are Instagram pages/profiles. Youtube videos are also the first thing that pops up when one is searching for some tutorials or, for example, cooking recipes.

IV. STATISTICS

We have put twenty-three of the most visited Serbian sites under validation [4]. The main purpose of this experiment was to find out whether their HTML and CSS codes are written by the standards or not. The most repeated errors will be described in the next few graphs. When it comes to the process of validation it was quite simple. We have selected the most recommended HTML validator (https://validator.w3.org/) and copied sites URL. Afterwards, all the errors and warnings were listed and put into our statistics, as shown in Fig. 3 and Fig. 4.

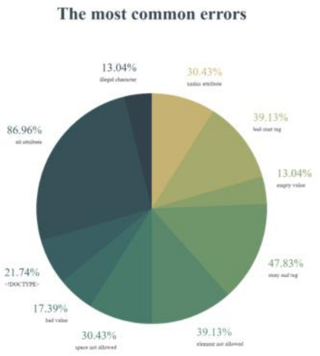


Fig. 3. Figure shows the most common errors during analysis

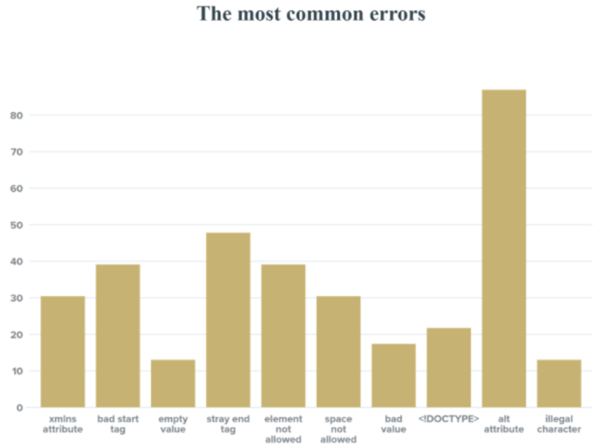


Fig. 4. Chart of the most common errors on Serbian websites

The next part represents the explanation of the top 10 most repeated errors.

1. *Xmlns attribute* - this error appears in quite many site codes, but the *xmlns* attributes are not allowed in meta tags.
2. <img> tag - cannot be in head. In that case, you should move it to body. Bad start tag in *img* in *head*.
3. Alt attribute - every *img* tag must have alt attribute
4. Stray end tag - in some codes, tags are not opened and closed properly
5. Element not allowed - element x not allowed as child of element y in this context. Depending on which element is being used, there are various types of context for each of those elements.
6. Space not allowed - in some tags, links have space between words, which is not allowed. But on the other hand, some tags do not have space between attributes and that is necessary.
7. Bad value - in this case the problem is that HTML tags cannot have CSS attributes. Value must be a digit.
8. <!doctype> - many codes do not have properly written *doctype* tag.
9. Empty value - element option without attribute label must not be empty.
10. Illegal character - when it comes to the *href* tags and values, some characters are not allowed.

V. CONCLUSION

Search Engine Optimization, now known as SEO, strives to improve sites' visibility on search engines. Altogether, our study showed many different SEO techniques for improving websites, of course, according to SEO standards and principles. We have strived to give quick and suitable/convenient suggestions and what is more important, easy understandable for ones who experience all the terms above for the first time.

The errors we have mentioned up above are not cardinal errors, but just very frequent. [7][8] Taking this into account, using validators, such as HTML and CSS validators, is something that should not be avoided. It is not only about the reduction of errors, but it also shows professionalism and provides guarantee that one's site will be valid in the future [9]. We recommend a few tools that we consider very useful. In the comment section of the article, one can find many tools, but we have selected some of the most interested [10]:

- SERP Volatility Tools - MozCast, Ayima Pulse etc.
- Forums and Blogs - Webmasters Help Community
- Google Update History - Google: Algorithm updates, Google Algorithm Updates History.

Generally speaking, while doing this research we have come to a few conclusions and restatement ideas. There are many factors for ranking on the web. As can be seen, it takes a lot of time and effort to become the first result. Although there is no guarantee for that, there are few steps that can be taken, that we have mentioned before. The right mix of off-page and on-page optimization techniques is extremely vital.

When it comes to question “Should one strive to please its audience or to please Google?

The answer is both. Each part of SEO needs to be in place in order to rank well. Advice: Create awesome content that users will love.

When it comes to experimental part of our work the conclusion we have come to, as has been noted, is that the most img tags do not have alt attributes. So that is the first one of our top three most repeated errors. If the site is not loading well, in that case, alt attributes are necessary. The example of code for this error is [5]:

```
<img src="img_girl.jpg" alt="Girl in a jacket"
width="500" height="600">
```

The second most repeated error is that some tags have “stray end tag”. The main problem is that tags are not opened and closed properly. To be precise, eleven out of twenty-three sites have this error.

Stray end tag noscript.

```
<noscript>
  <img height =”...” width=”...” src=”#”>
</noscript>
```

And finally, nine out of twenty-three sites have bad start in tags. For example, img tags or any other tag cannot have CSS attributes.

```
<img height=”1” width=”1” src=”#”/>
```

In this case, <img> also cannot have closed tag.

References

References should be typed in 8 pt text. Reference numbers should be indented and justified, like in the examples below (Style “References”). The reference numbers are on the line, enclosed in square brackets. In all references, the given name of the author or editor is abbreviated to the initial only and precedes the last name. Please write the names of *all* authors; do not use “et al”. Do not combine references: there must be only one reference with one number. If there is a URL, it can be included at the end of the reference. Always write full titles. Abbreviate the names of the journals. Examples of different kinds of references (journal articles, books, book sections, patents, etc.) are given below.

[1] SEO Made Simple: A Step-by-Step Guide for 2020 (online): https://neilpatel.com/what-is-seo

[2] Search engine optimization (online): https://en.wikipedia.org/wiki/Search_engine_optimization

[3] 10 Crucial SEO Ranking Factors You Need to Know (online): https://optinmonster.com/seo-ranking-factors

[4] Top Sites in Serbia (online): https://www.alexa.com/topsites/countries/RS

[5] HTML <img> alt Attribute (online): https://www.w3schools.com/tags/att_img_alt.asp

[6] Search Engine Optimization (SEO) Starter Guide (online): https://support.google.com/webmasters/answer/7451184?hl=en

[7] A. Elsadai, M. Tair, N. Savanović, “The Evaluation of The Technical Quality of Web Sites With Contents in Arabic and Serbian Languages,” in Sinteza 2019 - International Scientific Conference on Information Technology and Data Related Research, Belgrade

[8] Kittisak Chotikitpat, Prachyanun Nilsook, Sunantha Sodsee, Techniques for Improving Website Rankings with Search Engine Optimization (SEO)

[9] Why Validate? (online): https://validator.w3.org/docs/why.html

[10] Becoming Better SEO Scientists, Mark TraphagenJuly (online): https://moz.com/blog/becoming-better-seo-scientists-whiteboard-friday

[11] SEO inside Newsrooms: Reports from the Field Dimitrios Giomelakis, Christina Karypidou and Andreas Veglis

Source Code Quality Evaluation Using Network Science

Ivan Blažić, Marko Mišić, Zaharije Radivojević

***Abstract*— Structural quality of the source code has a great impact on the efficiency of software development processes. The cost and the complexity of adding functionalities or fixing a bug depend on how well-structured the code is. A standard way of measuring the structural quality of software is the calculation of code quality metrics, which are statistical data obtained by static source code analysis. This paper presents a software system for calculating code quality metrics by analyzing the network of dependencies between source code elements. Adaptations of well-known metrics can be calculated using this method, but with a more advanced application of network science new metrics can be introduced. Description of the applied method, the implemented system, as well as the results of a use case analysis are presented.**

***Index Terms*—Code quality metrics, network science, static source code analysis.**

I. Introduction

Code complexity rapidly increases the cost of performing changes to a software system, but also the risk of introducing bugs and errors [1]. For this reason, code maintainability becomes an important aspect of the software development process as a way of overcoming complexity issues. Maintainability is related to the effort needed to analyze, modify, or test a software system. It is usually achieved with a proper architecture and good coding practices that keep the components as simple and independent as possible [2]. This property is strongly affected by the structural quality of the software system.

Evaluation of the structural quality is usually done by calculating the code quality metrics, a collection of statistical data about the code structure [3]. Calculation of these metrics is commonly performed by simple static code analysis, which mostly consists of counting the number of elements and specific relationships, such as lines of code, number of classes, or number of methods called from a certain class. Common approach to source code quality analysis consists of combining traditional metrics in order to form a quality evaluation system [4][5].

Another approach to such analysis is the application of network science, the study of complex networks [6]. Software systems consist of code elements (e.g. classes, functions, or variables) and their dependencies (e.g. calls, inheritance, or variable access) which can be represented as a complex network [7]. The main benefit of the proposed network-based approach to source code analysis is the focus on the architecture and global topological properties of a software system as a whole, rather than a summary of local features [8]. For example, metrics based on advanced network analysis presented in the section IV imply the application of complex networks and cannot be calculated with a traditional approach.

In this paper, an approach to calculating code quality metrics by analyzing different types of dependency networks generated from the source code, as well as a software system that implements such approach, are presented.

The paper is divided as follows. The following section presents related work on the topic. A more detailed description of the dependency networks and applied approach is given in section III. Section IV describes used metrices, while functionalities of the implemented system are described in Section V. The results of use case code analysis are discussed in Section VI. Conclusion and directions for future work are given in the final section.

II. Related Work

Analysis of software systems using complex networks is an ongoing topic in the research community. This section provides a brief overview of the network-based approaches to software analysis applied in the related research.

Common application of network science is the study of social networks. Social network measure of the influence of an individual researcher called *h*-index has served as an inspiration to a new class centrality metric that has been introduced in [7].

Network-based measures have been used to predict defects on a program class or module level in several research papers [9][10][11]. Results have shown correlation between network properties and the number of defects on a class or module level. In some cases, the network-based measures can outperform the standard ones.

Structural quality measures based on network analysis have been introduced in previous research [12][13][14]. Network properties such as degree distribution, clustering, average path length, or modularity have been used to identify component cohesion, stability, complexity, and modular structure.

The related research is focused on the theory behind the proposed metrics, but few of them presents the technical implementation.

III. Dependency Networks

Code quality evaluation process presented in this paper consists of generating 3 types of dependency networks that are used to calculate 15 metrics. Each metric is calculated by

Ivan Blažić is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: blazic.ivan@outlook.com).

Marko Mišić is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: marko.misic@etf.bg.ac.rs).

Zaharije Radivojević is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: zaki@etf.bg.ac.rs).

analyzing one or two types of these networks. Values of all metrics are used to calculate a numerical grade for different aspects of quality. Final quality grade is the average of all sub-grades for individual quality aspects. The implemented software system allows the user to configure the influence of the individual metric to the quality grade and define aspects of quality.

The reason for using multiple types of networks is to separate the information needed for calculating different metrics. For simplicity, the following constraints are introduced in our approach:

- Focus is strictly on the software written in object-oriented paradigm.
- Only the source code written in Java language is supported.
- Annotations are ignored.
- Method implementations within interfaces are ignored, as this is considered as an improper use of interfaces.
- Dependencies to external libraries are ignored to avoid incomplete dependency networks.
- Built-in types and classes of Java standard library are ignored, as they considered to be a part of the language.
- All information is obtained by direct syntax-based analysis of explicitly defined dependencies. Therefore, any dependencies created by dynamic properties such as Java reflection or polymorphism are ignored. All implicit declarations such as inherited members or default constructors are not modeled.

All used networks are in a form of a directed simple graph. Each node, no matter the network type, has the following attributes:

- Identifier used in the code (e.g. class, method, or variable name).
- Type of the node (class, interface, method, field, or enum definition).
- Access modifier (public, private, protected, or default).

Links are commonly defined by a source node, a destination node, weight, and dependency information. Since two nodes can have multiple types of dependencies, link attributes are used to store this information. Self-links do not exist, therefore the source and destination nodes are always different. Types of source code dependencies covered in the presented approach are:

- Field type dependency (a class depends on all classes that are types of its members)
- Implementation (a class depends all on interfaces it implements)
- Inheritance (a class depends on its parent class)
- Inner class definition (a class depends on all inner classes it defines)
- Instantiation (a class depends on all classes that are types of objects it instantiates)
- Method call (a class depends on all classes that define methods it calls)
- Field definition (a class depends on its member variables and constants)
- Method definition (a class depends on its methods)

Depending on the actual source code dependencies they model, used networks are divided in the following 3 types.

A. Call graph

This network represents the dependencies between methods created by nested calls. Nodes represent the methods, and links represent calls. Direction of the links indicate the call hierarchy. This graph correlates to all method calls in the system.

B. Class structure graph

This network consists of several weakly connected components that represent each class. These components model the membership of fields (methods, constants, variables, and inner classes) to a class. Nodes can represent a class (central node), or different types of class members (peripheral nodes). Links represent field membership or a field usage in a method. Weight of the links between the central class node and a method node is the cyclomatic complexity [15] of the method.

C. Type dependency graph

This is a network of dependencies of any kind between each type (class or interface) in the system. Nodes represent types, and links are created between classes that have at least one of these dependencies between them. Figure 1 shows one part of a sample type dependency network.

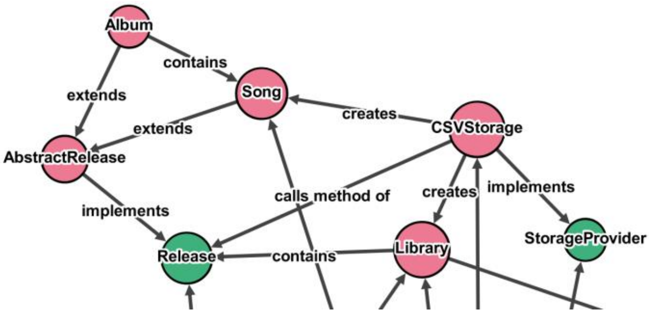


Fig. 1. Partial example of a type dependency network.

IV. USED METRICS

This section provides an overview of code quality metrics used in the presented approach. All metrics used in this paper are based on network analysis. To have a better overview of the contribution that network science brings to software quality analysis, applied metrics are divided to the following two categories:

A. Metrics based on simple network analysis

This category includes the adaptations of a well-known set of metrics [3][15][16]. Adaptations are made for some metrics to fit the network-based approach. The use of networks in this category of metrics comes down to identifying and counting nodes and links with certain properties. This set includes the following metrics:

1) *Interface Size (IS)*

Total number of publicly accessible members of the class. This metric is a simple measure of the complexity of interfacing a class. It is calculated as the degree of the class node in the class structure graph, by observing only public members.

2) *Number of Methods (NOM)*

Total number of methods in a class, calculated as a class node degree in the class structure graph, by observing the method node neighbors. Methods with overloaded signatures are modeled as separate methods.

3) *Weighted Method Count (WMC)*

Sum of cyclomatic complexities of all methods in a class. Cyclomatic complexity is not calculated with a network-based approach, but with a simple abstract syntax tree traversal and count of branching elements. The value of method complexity is embedded as the weight of the link between a class node and a method node. Therefore, this metric is calculated in the same way as NOM, but with using a weighted degree instead.

4) *Response for Class (RFC)*

Sum of the number of public methods and the number of methods called from those methods. This is calculated by identifying the public method nodes in the class structure graph and summarizing their count with their degree in the call graph.

5) *Number of Implemented Interfaces (NII)*

This is an adaptation of *Number of Superclasses* metric. Since Java does not support multiple inheritance, this metric can only be applied to implementing interfaces. Calculated as the degree of a class node in the type dependency graph, by observing only links that represent the implementation dependency.

6) *Coupling Between Objects (CBO)*

Total number of other classes a class is coupled with. Coupling in this context means having any kind of dependency besides inheritance between two classes. Calculated as the degree of a class node in the type dependency graph, with filtering out the implementation and inheritance dependency.

7) *Tight Class Cohesion (TCC)*

This is a class modularity measure that evaluates the degree of member re-use. It is the ratio between the number of pairs of methods that commonly use at least one field variable and the total number of pairs of methods in a class. Pairs of methods are identified by searching the class structure graph for method definition and field usage links, the rest of the calculation is a simple formula for this metric.

8) *Number of Variable Fields (NOVF)*

Total number of member variables declared in a class. Calculated as a degree of the class node in the class structure graph, by observing only the variable member nodes as neighbors.

9) *Number of Subclasses (NSUB)*

Total number of classes inherited from the observed class or interface. Calculated as the in-degree of the node in the inheritance graph.

B. *Metrics based on advanced network analysis*

This category represents a set of metrics that are based on network properties [7][17] of three modelled networks, such as betweenness centrality of the nodes, network path lengths, cycle count, etc. A network-based approach to calculating these metrics is either required or more convenient due to their definition. Metrics from this category rely on complex network properties instead of simple link or node counts. These are the following metrics:

1) *Depth of Inheritance Tree (DIT)*

The longest inheritance path in the system. A longest path in a tree is its depth. Inheritance graph is created by selecting inheritance and implementation links from the type dependency graph. As the inheritance graph consists of multiple trees, the value of this metric is the maximum of depths of all trees.

2) *Number of Circularly Dependent Classes (NCDC)*

Total number of classes that are involved in at least one circular dependency. Calculated by identifying all cycles in the type dependency graph and counting the number of classes involved in those classes.

3) *Number of Disconnected Groups (NDG)*

Total number of groups of classes that are disconnected from other groups in the system. In terms of network science, this metric is the number of weakly connected components of the type dependency graph. Ideally, the whole network is weakly connected and there are no disconnected groups. Calculated by evaluating the connectivity of the type dependency graph and counting the number of weakly connected components.

4) *Degree of Interdependency (DOI)*

Ratio between the total number of existing dependencies between classes and interfaces in the system and the theoretical maximum number of those dependencies. This metric measures the total density of dependencies between all classes and interfaces. Calculated as the density of type dependency graph.

5) *Maximal Call Indirection (MCI)*

Length of the longest call path in the system. Calculated as the longest path in the call graph.

6) *Decoupling Impact (DI)*

Ratio between the number of type dependency paths that pass through the observed class and the total number of those paths. From a network science perspective, this is the betweenness centrality of class node in the type dependency graph. This metric measures the degree in which the node acts like a bridge between different groups of nodes.

From the quality analysis aspect, metrics are divided in class-level metrics (IS, NOM, WMC, RFC, NII, CBO, TCC, NOVF, NSUB and DI) and system-level metrics (DIT, NCDC, NDG, DOI and MCI). All metrics except TCC and DI represent a negative measure of quality, which means that higher quality structures should have as low values of these metrics as possible. The reason for this is that they all measure complexity and size, which are not desirable properties of software components. High cohesion on the other hand is desirable, as it measures modularity and re-use. The impact of the value of DI metric on quality can not be generalized, because it depends on the expected role of the observed class. Observations made regarding the use of this metric shall be presented in the results.

V. Software system implementation

A software system that applies the described approach was implemented within this paper [18]. It is a configurable console application for Java 1.8 platform. Configuration of the system allows the user to define various analysis parameters by simply editing JSON files available within the system. If the user does not change any parameters, the default configuration remains in use. With the system configuration the user is free to decide how the metric values influence the quality grade and which metric values are considered ideal. The system reads the configuration files, takes a source code directory as input, and performs the following functionalities:

A. Generating and exporting dependency networks

All dependency networks described previously are generated based on the source code given as an input. One additional network is generated, which is the mix of all networks called *unified graph*. This network is not used for the purpose of quality evaluation, but for a visual representation of all dependencies. These networks are exported as CSV, DOT, GEXF or GML files and can be used for a custom user analysis of the networks using other network analysis tools and libraries.

B. Quality grade evaluation

A grading system was introduced to present the quality evaluation results more intuitively. The user defines a reference value *Mref* for each metric. A metric grade *Gm* is calculated based on the reference value and the real metric value *M* as shown in (1) and represents a quality grade from a single metric perspective. TCC metric is an exception where *Gm* value is always the actual value.

A quality property grade *P* is defined by a name and a weighted sum of selected metric grades. For example, “Architecture” property grade can be influenced by NCDC, NDG, DOI and DIT with respective weights. The weight of each metric grade for a specific property is defined by the user as the metric factor *F*.

Actual quality grades range from 0 to *Gmax* defined by the user. Quality grade for a single property is calculated as shown in (2), and the final quality grade is the average of all quality property grades.

$$G_m = \begin{cases} \left(\frac{M_{ref}}{M}\right), & M > M_{ref} \\ 1\,, & M \le M_{ref} \end{cases} \tag{1}$$

$$P = \sum_{i=1}^{N} (F(i) * G_m(i)) * G_{max} \tag{2}$$

C. Generating a quality analysis report

The results of the quality evaluation, based on the selected metrics, are presented in a form of a HTML report. This report contains the values of all metrics together with all grades and some additional information regarding the results (e.g. names of the classes involved in a circular dependency or names of the methods that make the longest call chain).

The system works by generating an abstract syntax tree (AST) of the source code, which is then traversed in order to create an abstract model of the code that later used when generating graphs, as shown on Figure 2. Each type of graph is created separately by iterating through the code model, filtering the elements and dependencies that are of interest for a specific graph, and converting this data into nodes and links. For example, the inheritance graph is created by extracting code elements that are represent classes or interfaces, and dependencies that represent inheritance or implementation. *Spoon* [19] library was used for generating the AST, and *JGraphT* [20] for working with graphs and networks.

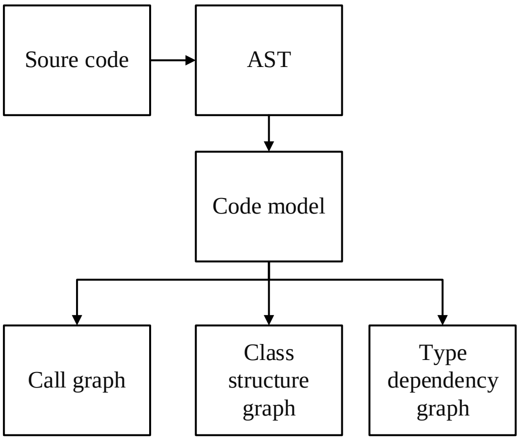


Fig. 2. Process of generating dependency networks from the source code.

VI. Results

All metrics covered in this paper are successfully implemented with the presented network-based approach. Each of the implemented metrics are covered with several automated test cases to verify their accuracy. This section presents a use-case of the approach, where the source code of the implemented software system is analyzed. Metrics that had the highest impact on the quality grade are presented, as well as the interpretation of DI metric. A more advanced analysis has been done for the exported type dependency network, and the results of this analysis are presented and discussed at the end of this section.

A. Analysis of metric results

Default system configuration was used in the presented use-case. In case of the source code of the implemented system NCDC, MCI and NDG metric values had the greatest impact on the quality grade. Table 1 presents the metric grades *Gm* for each system-level metric, as well as their actual and reference values. As described in section V, a metric grade is calculated based on actual and reference value of a metric. Reference values are specified in the system configuration. Class-level metrics did not have a significant impact on the quality grade.

TABLE I
Values of system-level metrics

Metric	Reference value (Mref)	Actual value (M)	Metric grade (Gm)
NCDC	0	30	0
MCI	4	10	0,4
DIT	5	2	1
DOI	0.3	0.025	1
NDG	0	2	0

B. Interpretation of DI metric

The impact of betweenness centrality on the structural quality of the source code cannot be generalized. This metric is a measure of separation between distinct groups of nodes. Depending on the predicted position of the node (class or interface), a high value of this metric may or may not be desirable. For example, utility classes that are accessed from many different parts of the code or interfaces that are intended to separate two bigger components are expected to have a high DI. Since these are special architectural roles, in most cases high DI values are not desirable.

Several classes in the implemented system have unexpectedly high DI, twice the average value or more. The cause of such values is the direct dependency to classes that normally have a high DI. In this way the observed class also has a high DI due to transitive dependencies inherited from another node.

C. Advanced analysis of the type dependency network

Properties of the exported network of type dependencies have been additionally analyzed outside of the implemented software tool. This analysis was done with Gephi [21] with the goal to study the community structure of the network as well as to examine the real-world properties.

1) Community structure

Community detection algorithm with default parameters of Gephi was applied to the network. Visualization of the results is shown in Figure 3, where a strong community structure can be observed. Nodes are grouped and colored by their association to communities.

The detected communities strongly resemble the relationship between classes in the same package or functionality. Amount of correlation between the community structure and source code module separation can indicate architectural flaws. Two examples are observed in the given network.

A certain node has been identified to strongly belong in the community of another package. This can be caused either by the lack of an appropriate interface or a mistake in functional decomposition.

Software modules should be strongly decoupled, which should reflect in strongly separated communities. Not all communities in the presented network are clearly separated, which indicates weak module decoupling.

2) Real world properties

Networks that model real systems are expected to have special properties comparing to ones that are random or trivial [22]. These properties include a strongly expressed community structure, a short average path length relative to the network size, and a *power-law* degree distribution that resembles the distinction of central and peripheral nodes that can be seen in preferential attachment network models.

Degree distribution of type dependency graph of the analyzed system is shown in Figure 4. Network diameter is 10, while the average path length is 4.03. All those properties of the presented network clearly show that it resembles real-world networks.

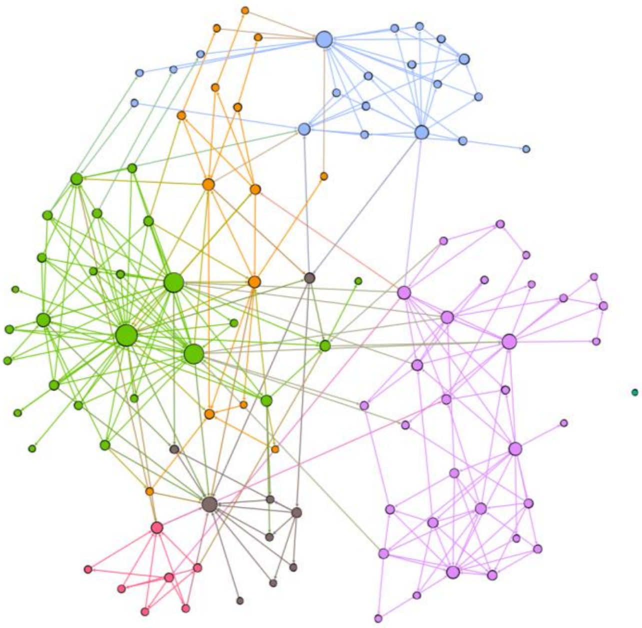


Fig. 3. Visual representation of the type dependency network of the implemented software system.

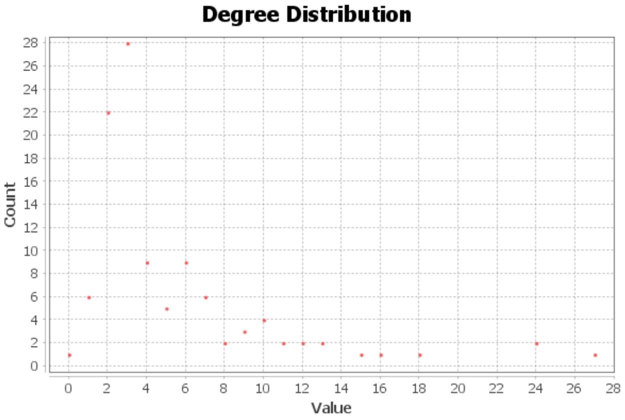


Fig. 4. Degree distribution of type dependency graph of analyzed system.

VII. Conclusion

This paper presents a software structure quality evaluation method based on the analysis of dependency networks generated from the source code. Three types of dependency networks were used to calculate a set of code quality metrics. Several well-known metrics have been adapted for this approach, and new network-based metrics were introduced. A software system that applies this approach has been implemented. A use-case of the implemented system, as well as an advanced analysis of one of the dependency networks are presented in the results.

Proposed approach has been successfully implemented and tested with the introduced software system. Additional examination of the properties of one generated network provided more information about architectural flaws and real-world properties of the source code. The downside of the proposed approach is the additional complexity of analyzing dependency networks that is not necessary for calculating simple metrics. This additional complexity also implies a lower calculation performance comparing to the traditional approach.

Implemented software system can be improved to support fine-grained dependency networks on the lexical level, such as control and data dependency graphs. Support for other programming languages would further improve the usability of the system. Topics of future research would include additional network-based metrics, such as class instability, clustering coefficient, and modularity. Another approach to future research would be the study of correlation between dependency network properties and reliable quality measurements.

ACKNOWLEDGMENT

This work has been partially funded by the Ministry of Education, Science, and Technological Development of the Republic of Serbia. Grant numbers III44009 and TR32047.

REFERENCES

[1] V. Antinyan, M. Staron, A. Sandberg, "Evaluating code complexity triggers, use of complexity measures and the influence of code complexity on maintenance time", *Empirical Software Engineering*, vol. 22, no. 6, pp. 3057-3087, Mar, 2017.

[2] R. C. Martin, *Clean Architecture: A Craftsman's Guide to Software Structure and Design*, Prentice Hall, Upper Saddle River, New Jersey, US, 2017.

[3] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design", *IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476-493, Jun, 1994.

[4] H. Washizaki, R. Namiki, T. Fukuoka, Y. Harada, H. Watanabe, "A Framework for Measuring and Evaluating Program Source Code Quality", Proc. Product-Focused Software Process Improvement, 8th International Conference, PROFES, Riga, Latvia, vol. 4589, pp. 284-299, Jul. 2-4, 2007.

[5] F. Madou, M. Agüero, G. Esperón, D. López De Luise, "Software for Improving Source Code Quality", *World Academy of Science, Engineering and Technology*, no. 59, pp. 1259-1265, 2011.

[6] National Research Council, *Network Science*, The National Academies Press, Washington, DC, US, 2005.

[7] Y. L. Ding, , B. He. Peng, "An Improved Approach to Identifying Key Classes in Weighted Software Network", *Mathematical Problems in Engineering*, vol. 2016, pp. 1-9, Jan, 2016.

[8] W. Pan, "Applying complex network theory to software structure analysis", *World Academy of Science, Engineering and Technology*, vol 60, pp 1636-1642, Dec, 2011.

[9] G. Concas, M. Marchesi, A. Murgia, R. Tonelli, "An Empirical Study of Social Networks Metrics in Object-Oriented Software", *Advances in Software Engineering*, vol. 2010, pp. 1-21, Jan, 2010.

[10] T. H. D. Nguyen, B. Adams, A. E. Hassan, "Studying the impact of dependency network measures on software quality", Proc. 2010 IEEE International Conference on Software Maintenance, Timisoara, Romania, pp. 1-10, Sep. 12-18, 2010.

[11] M. Orrù, C. Monni, M. Marchesi, G. Concas, R. Tonelli, "Predicting Software Defectiveness through Network Analysis", Proc. Seminar On Advanced Techniques and Tools for Software Evolution, Mons, Belgium, vol. 1820, pp. 36-47, Jul. 6-8, 2010.

[12] M. Savić, Ivanović Mirjana, "Graph clustering evaluation metrics as software metrics", Proc. 3rd Workshop on Software Quality Analysis, Monitoring, Improvement and Applications, Lovran, Croatia, vol. 1266, pp. 81-89, Sep. 19-22, 2014.

[13] L. Šubelj, M. Bajec, "Software systems through complex networks science: review, analysis and applications", Proc. First International Workshop on Software Mining, Beijing, China, pp. 9-16, Dec., 2012.

[14] W. Pan, C. Chai, "Measuring software stability based on complex networks in software", *Cluster Computing*, vol. 22, no. 2, pp. 2589-2598, Mar, 2019.

[15] T. J. McCabe, "A Complexity Measure", *IEEE Transactions on Software Engineering*, vol. SE-2, no. 4, pp. 308-320, Dec., 1976.

[16] J. M. Bieman, B-K- Kang, "Cohesion and reuse in an object-oriented system", *ACM SIGSOFT Software Engineering Notes*, vol. 20, no. SI, pp. 259-262, Aug, 1995.

[17] T. D. Oyetoyan, "Dependency cycles in software systems: quality issues and opportunities for refactoring.", Ph.D. dissertation, NTNU, Faculty of Information Technology, Mathematics and Electrical Engineering, Department of Computer and Information Science, Trondheim, Norway, 2015.

[18] I. Blažić, "Code Quality Analysis Using Network Science", Source code, https://github.com/BlazicIvan/Net-CQA, 2020.

[19] R. Pawlak, M. Monperrus, N. Petitprez, C. Noguera, L. Seinturier, "Spoon: A Library for Implementing Analyses and Transformations of Java Source Code", *Software: Practice and Experience*, vol. 46, no. 9, pp. 1155-1179, Sep, 2016.

[20] D. Michail, J. Kinable, B. Naveh, J. V. Sichi, "JGraphT—A Java Library for Graph Data Structures and Algorithms", *ACM Transactions of Mathematical Software*, vol. 46, no. 2, Article 16, Jun, 2020.

[21] M. Bastian, S. Heymann, M. Jacomy, "Gephi: an open source software for exploring and manipulating networks", Proc. International AAAI Conference on Weblogs and Social Media, San Jose, California, May 17 - 20, 2009.

[22] R. Albert, A-L. Barabási, "Statistical Mechanics of Complex Networks", *Reviews of Modern Physics*, vol 74, no. 1, pp 47–97, Jan, 2002.

Android Application For Promoting The Cultural Institutions Through Wi-Fi Access

1st Vladislav Tokovic
Technical Faculty
Singidunum University
Belgrade, Serbia
vladislav.tokovic.19@singimail.rs

2nd Luka Gajic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
luka.gajic.17@singimail.rs

3rd Dusan Cvetnic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
dusan.cvetnic.17@singimail.rs

4th Miodrag Vidacic
Technical Faculty
Singidunum University
Belgrade, Serbia
miodrag.vidacic.19@singimail.rs

5th Nikola Savanovic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
nsavanovic@singidunum.ac.rs

6th Miodrag Zivkovic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
mzivkovic@singidunum.ac.rs

***Abstract*—Tourists frequently use available wireless connections, to keep their roaming costs as low as possible. Such wireless connections exist in hotels, bars, restaurants, and similar places tourists frequently visit. Work presented within this paper proposes another approach for providing wireless internet connection, both for tourists from abroad and the citizens of the country. We have aimed to develop a mobile app that combines enjoyable experience - good internet connection and informative experience – providing cultural and historical information about countries' landmarks interactively and entertainingly. In this way, it is possible to achieve the promotion and marketing of cultural institutions, such as museums, libraries, churches, monasteries, art galleries.**

Keywords— mobile application, Android, Wi-Fi, tourism, marketing

I. Introduction

The global Internet usage (Internet usage is the number of people who use the internet worldwide [1]) has followed the trend of the constant increase. Consequently, global Internet usage, at this moment, is higher than ever before, and there are no indicators that the current trend will be abandoned. Some reasons for that are an increase in Internet service availability, technology improvement, and initiatives to provide internet access to people in impoverished areas.

Another demand that exists simultaneously with internet usage, in both consumer and business markets, is the need for high bandwidth. The need for a high rate of data transfer has been present ever since people have started interconnecting computers. However, the adjective "high" is relative to the time we refer to. The rate of data transfer that was once considered decent would now be deemed unusable. The increase of computer and data complexity has been very closely followed by the necessity for higher bandwidth, and thus far that necessity has been fulfilled- the advancement of interconnection technology has also been continual. That is very noticeable in the comparison of copper and fiber-optic cables, which is the most recent technology leap in data transfer through the computer network. Fiber-optic cable is superior, in some metrics by the order of magnitude, in virtually every relevant metric, compared to copper-wire cable: bandwidth, transmission speed, and distance, reliability, and security. Currently, the only metric where fiber-optic technology is lagging behind copper-wire technology is the price, but that gap is also continually shrinking.

So far, we have described the state of matter in the context of wired internet connection and personal computers. The number of mobile phones has long surpassed the number of personal computers [2], and concurrently, there has been a need for the higher bandwidth of the wireless connection. The most notable recent technology advancement is the development of a 5G mobile network. 5G connection can provide data rate transfer comparable to the fiber-optic cable similarly, again conditionally, the bandwidth of the 4G connection is comparable to copper-cable connection bandwidth. However, the cellular connection depends on the infrastructure of the internet service provider-proximity of the cell towers.

It makes sense to use mobile network connection when the user's movement is not limited to the area of single/local wireless connection, for example during the commute or when there are no available wireless connections.

Internet connection provided through some number of wireless routers located inside residential or non-residential (business, institutional, educational, and similar) building is limited to space inside and in the proximity of that building. That is why in the office or home environment, when the occupants of the building spend substantial continuous portions of the time in the building area, it makes sense to use a wireless router connection. Naturally, the amount and arrangement of routers should be selected adequately, to accommodate the building's needs for a stable connection. In the mentioned settings it could be concluded that wireless router connection is preferred to mobile network connection, however, there are two scenarios where wireless routers are a necessity:

1. In the areas with bad mobile phone service

2. For tourists visiting abroad, were roaming cost could be significant.

The rest of the paper is organized as follows. In section two, we give the motivation and the overview of the state of the art solutions in the considered domain. In section three, we describe the proposed implementation of the Android application. In section four, we describe the backend implementation. Finally, section five gives concluding remarks and future work.

II. Motivation and Background

Tourists can usually use the wireless connection on the premises of their accommodation, where they are provided with login information (connection name and password). Similar options are given inside most coffee shops or restaurants that people visit. Sometimes, in return for having access to the wireless connection, people are required to log in using Google or Facebook accounts, in other words, trade some of the personal info for wireless connection. In all the mentioned options, internet access is in a sense commercialized, either by providing added value to the experience, like in hotels (although internet connection at this point could be considered almost basic need) or more directly by collecting consumer data. This is justified because the installation and maintaining a local wireless network represents a monthly expense for the business.

The application of mobile technologies in tourism is not a new concept. Some early papers on the topic anticipated the huge potential tourism offers for the use of the new mobile technologies [3]. The first applications were mostly in the form of electronic guidebooks, maps, and tour guides [4]. More recently, focus moved to the personalized recommendation services which suggest products and offer different services based on the location of the user [5] Location-aware devices, together with the further development of the web, enabled the creation of the hybrid applications that can offer personalized context-aware and location-aware services to the users [6]. Location-aware service focus on providing the information and services to the user based on the user's current position. Mobile applications today are considered to be the future of the tourism industry, as stated in [7].

On the other hand, new technologies offer new ways to understand complex consumer behavior. The data obtained through Wi-Fi services available in restaurants and coffee shops made it possible to acquire in-depth knowledge of consumer behavioral patterns. One such study [8] analyzes the impact of free Wi-Fi access on customer loyalty. Governments and city authorities worldwide are installing hotspots across frequently visited places (typically touristic attractions) to connect their citizens to information and other online services. At the same time, businesses are attracting and retaining customers with value-added free Wi-Fi services on their premises, as stated in [9].

Our work in this paper considers another viable approach for providing wireless internet connection, targeted at tourists from abroad, but also citizens of the country. We have aimed to develop a mobile app that combines enjoyable experience - good internet connection and informative experience – providing cultural and historical information about countries' landmarks in an interactive and entertaining manner. The promotion and marketing of cultural institutions, such as museums, libraries, churches, monasteries, art galleries is not an easy task. Some aspects of culture cannot be profitable and should not be profitable, and as such, private and government funding is necessary. Investing in that sense comes with the reality that there will not be a material return on that investment, because investing in culture means enlightening and culturally uplifting society. The return on that kind of investment is the increased intellectual and social wellbeing of the whole society. Culture is the only sector that can provide such an all-encompassing and non-discriminatory effect because it should exist and be welcoming for all people, regardless of a demographic or social group. However, as it is with other types of investing, investing in culture should be smart, tactical, and utilizing all the merits of new technologies. Embracing digitalization and using it to its benefit would undeniably put the cultural sector on a good path towards fulfilling its inherent purpose – serving as societies' omnipresent intellectually, creatively, emotionally, and socially enhancing and elevating experience.

III. Mobile Application

This Android application is in progress and it is available only through an emulator. This APP will be available on Google Play Store. Behind the application on the back end side there are database, web server and Android OS. It was decided that the database would be based on the relational model, using MySQL RDBMS. Android version is Nexus 5X API 29. Server's main interface oriented to the mobile application is designed using REST (Representational State Transfer) architectural style. As for the used technologies in which the application layer is implemented, the authors used Node.js server-side platform based on JavaScript programming language. The application is hosted online using Amazon Web Services Elastic Beanstalk service. The main idea of downloading applications is that the user is able to download IT scanning QR code. For example, if the end user is a tourist it would be recommended to download application in his/her country. The second option is that on airport in Serbia there is one info desk, where there is an available wifi hotspot only for that purpose. Navigation in this software would be downloaded and a live map should be available through the GPS system. Applying GPS system the user has completely free navigation to specific tourist and cultural destinations in Serbia.

We have realized the idea described in this paper in the form of an Android mobile application using Android Studio IDE.

Our goal was to develop application functionalities that support and reflect the core idea of this service, and not to devote much time, at this point, toward improving the visual design of application and user experience. We have also provided screenshots of the application to complement the written part of the paper. Key functionalities of applications are presented in this section.

1. On application launch, a basic google map is displayed, and the map is zoomed and focused on the current user location. Concurrently, the application makes an asynchronous GET request, getting as a response array of geographical coordinates of the cultural monuments with available wireless connections. Areas, where the wireless internet is available, are marked on the map with red circles, as shown in Fig. 1.

2. If the application user is not in the vicinity of available connections (user/device marker on the map is not located inside of any of the red circles) nothing else is displayed on the screen with the map. The mobile application is getting periodical updates on device location, all location changes are displayed on the map, and after each location update, the application checks whether the device is in the wireless connection area, as shown in Fig 1.

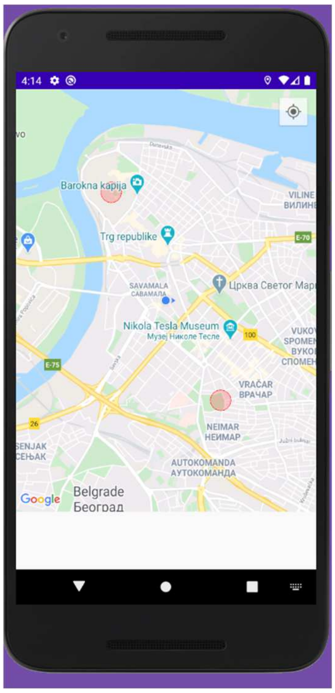


Fig. 1. Cultural monuments marked with red on the map

3. When the user enters the wireless internet area, or if the user was already in that area on the application launch, the "connect to Wi-Fi" button is made visible on the screen, as shown in Fig. 2. That button is only present on the screen while the user is inside the internet area. Simultaneously, Another GET request is made, and as a response, the array containing quiz questions and possible responses are received. When the user presses the button, the quiz, in the form of several radio groups – one radio group per question, is displayed.

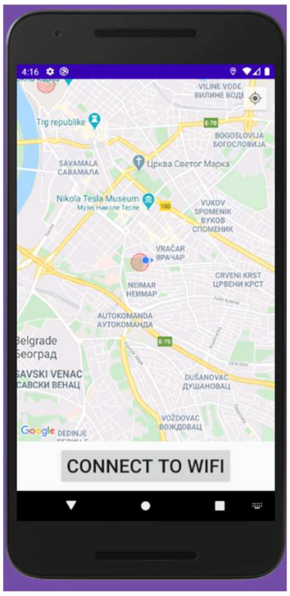
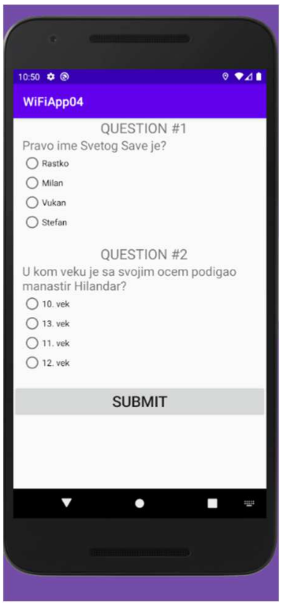


Fig. 2. Option to connect to the Wi-Fi is triggered when the user is in the proximity of the target area.

4. The user should then complete the quiz about the cultural or historical monument at his location and submit the answers by pressing the "submit" button at the bottom of the form, as shown in Fig. 3. The application validates the quiz form, and if any part of the quiz was left unanswered, the user is notified that all questions must be answered and lists the unanswered questions.


10:50

WiFiApp04

QUESTION #1

Pravo ime Svetog Save je?

○ Rastko
○ Milan
○ Vukan
○ Stefan

QUESTION #2

U kom veku je sa svojim ocem podigao manastir Hilandar?

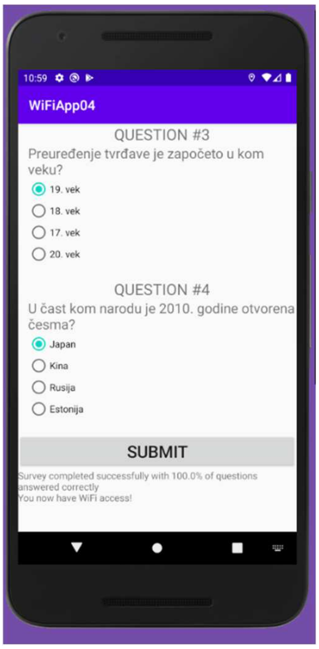
○ 10. vek
○ 13. vek
○ 11. vek
○ 12. vek

SUBMIT

Fig. 3. A quiz about the cultural monument on the user's location

5. When the whole quiz is finished and submitted, the POST request is made, sending the answers selected by the user. As the request response, the application receives the percentage of questions that were answered correctly, as shown in Fig. 4. Depending on the percentage, the user is either notified that they have internet access, or that they do not have internet access because of the insufficient number of correct answers, in which case they can retake the quiz.

The final part of the application, which is actually giving the Wi-Fi access to the user if the quiz has been successfully completed, is still under development.


10:59

WiFiApp04

QUESTION #3

Preuređenje tvrđave je započeto u kom veku?

● 19. vek
○ 18. vek
○ 17. vek
○ 20. vek

QUESTION #4

U čast kom narodu je 2010. godine otvorena česma?

● Japan
○ Kina
○ Rusija
○ Estonija

SUBMIT

Survey completed successfully with 100.0% of questions answered correctly
You now have WiFi access!

Fig. 4. After the quiz is completed, results are shown to the user, and option to turn on the Wi-Fi will be opened

IV. Backend Application

In this section, we describe the implementation of the backend side of the proposed Wi-Fi application.

A. Database

The first thing that was needed in the process of successfully implementing back-end logic was database design and working with test data. It was decided that the database would be based on the relational model, using MySQL RDBMS, which is a highly popular solution for the implementation of relational databases. The model itself is based on the main entity – location. Because the application works with specific historical locations in the city, it is needed to store several things about the location:

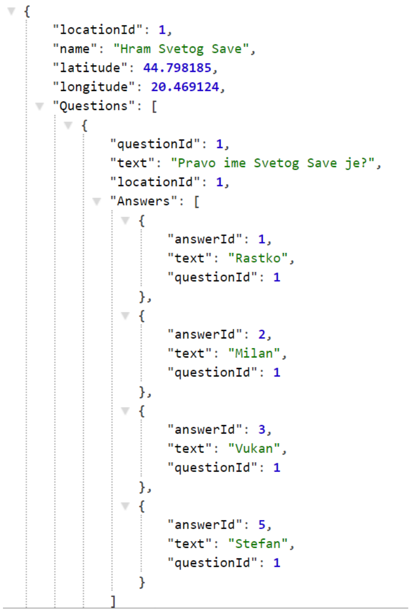
- Name – User can easily see which location is closest to him

- Latitude and longitude – Using these two parameters, the application can get questions for the user to answer based on the closest known location

The next two important entities are question and answer. Entities are related in such a way that one location can store multiple questions, and one question can have multiple answers where only one answer can be correct for that question (single-choice questions). The database itself is hosted online using the Amazon Web Services cloud computing platform, more precisely its' RDS (Relational Database Service) system.

B. Web Application

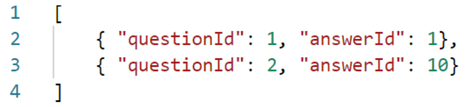
Because the main part of the application is based on mobile technologies (Android) there needs to be an established connection between the mobile application and remote server, as well as between server and database. The server is in the duty of preparing data fetched from the database, based on what the user has requested. Its' main interface oriented to the mobile application is designed using REST (Representational State Transfer) architectural style, which has some specific constraints such as URI identification of resources and statelessness (it doesn't share the same application state during multiple requests, each request is independent). Web application exposes several specific endpoints and using these endpoints applications can query data that's needed at that moment, using GET HTTP methods. Also, there is an endpoint using the POST HTTP method where the application can send answered questions for the specific location and as a response it gets JSON (JavaScript Object Notation) serialized object with status message and percentage of correctly answered questions, as shown in Fig. 5.



```
{
    "locationId": 1,
    "name": "Hram Svetog Save",
    "latitude": 44.798185,
    "longitude": 20.469124,
    "Questions": [
        {
            "questionId": 1,
            "text": "Pravo ime Svetog Save je?",
            "locationId": 1,
            "Answers": [
                {
                    "answerId": 1,
                    "text": "Rastko",
                    "questionId": 1
                },
                {
                    "answerId": 2,
                    "text": "Milan",
                    "questionId": 1
                },
                {
                    "answerId": 3,
                    "text": "Vukan",
                    "questionId": 1
                },
                {
                    "answerId": 5,
                    "text": "Stefan",
                    "questionId": 1
                }
            ]
```

Fig. 5. An example of an application endpoint for fetching questions based on a specific location

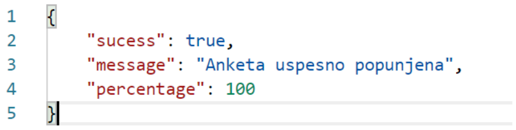
When the mobile application queries questions for a defined location, the response is first prepared on the server-side, then serialized using JSON, and finally returned through the HTTP response. There is no indication which of the following answers is correct as it's information known only by the server in the process of checking answered questions, as shown in Fig. 6.



```
[
    { "questionId": 1, "answerId": 1},
    { "questionId": 2, "answerId": 10}
]
```

Fig. 6. An example of prepared answers for POST HTTP request

When answering questions, the user is expected to provide an array of objects, each containing question ID and answer ID of what the user thinks is the correct answer, as shown in Fig. 7.



```
{
    "sucess": true,
    "message": "Anketa uspesno popunjena",
    "percentage": 100
}
```

Fig. 7. An example of the finished quiz with the percentage of correctly answered questions

As for the used technologies in which the application layer is implemented, the authors used Node.js server-side platform based on JavaScript programming language. The main advantage of this platform is speed, as well as non-blocking, asynchronous code execution based on events. It is widely supported by many cloud computing service providers, which makes the whole process of deployment much easier. The application is hosted online using Amazon Web Services Elastic Beanstalk service, which is used for easier code deployment and supports many popular server-side technologies (Node.js, Java, .NET, PHP).

V. Conclusion

We have developed this version of the Android application as a pilot project, with only presentable functionality. There are two major distinct applications based on the same concept of wireless internet connection as “reward“ for users that we plan to pursue in the future. Described below are planned functionalities for both versions:

1. Not-for-profit application (culture and entertainment-oriented, consumer-oriented):

In order to expand on and improve the application, assistance, and funding from different institutions and government subjects would be necessary. One approach would be to offer the project as a type of private-public partnership to major subjects involved in the process of digitalization in Serbia: The Ministry of Public Administration and Local-self government, The Ministry of Trade, Tourism and Telecommunications and The Chamber of Commerce and Industry of Serbia. The process of digitalization in Serbia is financed through different funding lines such as EU funds (IPA, European Investment Bank, Horizon 2020) or bilateral cooperation program (GIZ).

The other approach would be towards the main public authorities in the cultural sector, on national, regional and local levels such as The Ministry of Culture and Information, The Secretariat for Culture and Public Information AP Vojvodina, The Secretariat for Culture – City of Belgrade or in direct collaboration with the particular cultural institutions (museums, cultural centers, archives, libraries). There is a network of more than 533 public cultural institutions operating in Serbia. In that case, funding would be provided through public calls and specific tenders. The disadvantage of this option would be the relatively small funds that would be available for the product development in this way

2. For-profit application (business and advertisement oriented):

The target group of this application will be people (both citizens and tourists) who only wish to have access to the continuous wireless internet connection in larger urban city areas. In order to have that service, they would have to create an application account and periodically answer short surveys and provide personal preferences about different consumer products, services, and brands. Based on the information the user provides, they would periodically receive advertisements and notifications about discounts and offers. One important aspect of data they share would be the location. For example, the information about specific brand shops they visit in shopping zones or pastry shops in pedestrian zones would allow for more efficient modeling of consumer behavior and preference analysis, with the need for fewer surveys. Such an application would have to be developed in cooperation with corporate entities that already have a complete or large portion of the network infrastructure needed. That is why the idea is to create this application as a joint effort with some internet service provider (ISP), who has substantial wireless internet area coverage, and as a team present this idea to businesses that do not utilize services of this ISP, in which case ISP would get more corporate users, and those businesses would get advertising space (application) and raw consumer data. Also, ISP could offer this as an additional package their existing business customers, with both business partners benefiting.

References

[1] Wikipedia, Global Intenret usage (online): https://en.wikipedia.org/wiki/Global_Internet_usage

[2] Statista.com, Computing device shipments forecast worldwide from 2013 to 2020 (online): https://www.statista.com/statistics/265878/global-shipments-of-pcs-tablets-ultra-mobiles-mobile-phones/

[3] B. Brown, M. Chalmers, “Tourism and mobile technology”. In: Kuutti K., Karsten E.H., Fitzpatrick G., Dourish P., Schmidt K. (eds) ECSCW 2003. Springer, Dordrecht, pp 335-354, 2003.

[4] M. Kenteris, D. Gavalas, and D. Economou, “An innovative mobile electronic tourist guide application”. Pers Ubiquit Comput 13, 103–118 2009.

[5] CC. Yu, H. Chang, “Personalized Location-Based Recommendation Services for Tour Planning in Mobile Tourism Applications”. In: Di Noia T., Buccafurri F. (eds) E-Commerce and Web Technologies. EC-Web 2009. Lecture Notes in Computer Science, vol 5692. Springer, Berlin, Heidelberg, 2009.

[6] A. B. Barragáns-Martínez, and E. Costa-Montenegro, "Adding personalization and social features to a context-aware application for mobile tourism." Hospitality, travel, and tourism: Concepts, methodologies, tools, and applications. IGI Global, pp. 467-480, 2015.

[7] Tan, Garry Wei-Han, et al. “Mobile applications in tourism: the future of the tourism industry?”, Industrial Management & Data Systems, 2017.

[8] A. Reyes-Menendez, et al. "Understanding the influence of wireless communications and Wi-Fi access on customer loyalty: a behavioral model system.", Wireless Communications and Mobile Computing 2018.

[9] F. Humairaa, A. Gokhool, and Y. Chuttur. "A Crowdsourced Mobile App Architecture for Plotting User Rated Free WiFi Hotspots." 2019 Conference on Next Generation Computing Applications (NextComp). IEEE, 2019.

Different Approaches in Serbian Language Parsing using Context-free Grammars

Teodora Đorđević and Suzana Stojković, *University of Niš, Faculty of Electronic Engineering*

Abstract—**Syntax analysis is an extremely significant phase of natural language processing. This paper presents a comparison of two methods for syntax analysis of the Serbian languages based on context-free grammars. First, it describes building a POS tagger with corpora. Secondly, it defines a context-free grammar for the Serbian language. After that, it explains how the syntax parser is created and compares its performance with a parser implemented using NLTK. Finally, it explains the post-processing layer which is added in order to reduce the number of syntax trees generated due to grammar ambiguity. The experiments showed that the implemented parser is on average 6115 times faster than the NLTK parser and that the post-processing reduced the number of syntax trees by 54% on average.**

Index Terms— **NLP; POS tagging; Context-free grammars; Syntax analysis; CYK algorithm**

I. Introduction

Natural language processing has been an essential theme in computer science and engineering since its very inception as evidenced by the postulation of the Turing test which was first described by the renowned computer scientist and visionary Alan Turing in 1950. Serving as the *lingua franca* of the modern age the English language has been the most heavily researched language in the NLP context. That being said, many of the techniques developed for processing English cannot be directly applied to many different languages because of certain fundamental differences in language structure. Each language has its own set of peculiarities that need to be taken into consideration when developing computer programs capable of comprehending them.

Syntax analysis or parsing is the process of analyzing a sentence in natural languages conforming to the rules of a grammar. Creating a successful syntax analyzer grants a good foundation for building more sophisticated semantic analyzers. Syntax analysis has its own role in Information Extraction, Question Answering systems, Rule-based Machine Translation, etc.

This paper presents a comparison between two parsers for the Serbian language, one developed by using NLTK [1] and the other implemented using the CYK algorithm [2]. Before implementing a syntax parser, it is necessary to train a POS tagger and then define a Context-free grammar for the Serbian language. The next step after completing the parsers is adding a post-processing layer in order to minimize the number of syntax trees and to remove the trees that are completely inconsistent with the language.

II. Related Work

Syntax analysis is a process of generating syntax trees for an input sentence. By applying syntax analysis, a sentence is given specific structure. There are certain approaches in syntax analysis or parsing. First, there is parsing with context-free grammars. A context-free grammar [3] represents a list of rules which are used to generate a syntax tree for a given input. This approach has various problems due to inability to predict every possible sentence structure. Also, adding more rules can generate more syntax trees and not lead to parsing improvements. The second group of algorithms is based on statistics. The first approach can generate a large amount of syntax trees, so adding statistics [4] is the best solution for this problem, because it helps determine which syntax tree is most likely to be correct. Before explaining algorithms for syntax analysis, it is necessary to present search methods that can be used. There are two strategies, top-down and bottom-up.

The top-down strategy is also called goal-directed search. This technique does not consider input itself, as it only considers whether a syntax tree can be generated from a given grammar. The top-down method starts with the starting symbol S and moves to the bottom. It is necessary to find all the rules that have the starting symbol on the left side and generate syntax trees with the starting symbol as the syntax tree root. After that, those constituents are used to expand the tree even more until the leaves are reached. At every level, the algorithm considers rules with the current symbol on the left side in order to expand a syntax tree. When the syntax tree with leaves is generated, it is necessary to rule out all the rules where the input sentence does not match the created syntax tree. Syntax trees that are not ruled out in this phase are the result of syntax analysis.

The bottom-up parsing represents a different approach. The parser starts from the input sentence, and tries to build syntax trees from input words, by going up. The parser attempts to move up the branch for each potential syntax tree by attempting to match the right side of the rule with the existing nodes. The rule that is a match is reduced to its left side until the starting symbol is reached. The parsing is successful if a syntax tree for a given input sentence exists and contains the starting symbol S as its root.

Each of these techniques has both its own advantages and disadvantages. The top-down method does not attempt to

Teodora Đorđević is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: teodora.djordjevic@elfak.ni.ac.rs).
Suzana Stojković is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: suzana.stojkovic@elfak.ni.ac.rs).

create syntax trees that do not have the starting symbol as its root, but also generates trees that are not a match to a given input. The bottom-up method does not generate any syntax trees that do not match the sentence being parsed, but it does explore trees that do not begin with the starting symbol and are thus invalid.

By using either one of the aforementioned methods parsing can be implemented effectively. The first algorithms designed for syntax analysis of natural languages used context-free grammars as a set of rules for parsing. This approach requires creating a context-free grammar and then choosing an appropriate parsing algorithm to generate syntax trees. When choosing a parsing algorithm, it is necessary to consider the search method. The algorithms for parsing using a context-free grammar-based approach are CKY, the Earley algorithm [5] and Chart parsing [6]. This paper will later explain an implementation of the CYK parser.

Parsing with context-free grammars can result in generating multiple trees for a single sentence. This ambiguity problem in most cases cannot be solved by adding more grammar rules. It can be solved by adding statistics which leads to a different approach – statistical parsing. Statistical parsing calculates probabilities for every possible syntax tree and chooses a syntax tree with the largest probability. A possible way of assigning these probabilities to the syntax trees is by using a Probabilistic Context-free grammar (PCFG) [7]. In PCFG, the CFG rules are extended with an associated probability which determines how likely it is that that specific rule will be applied.

III. POS TAGGING

Tagging is an essential part of natural language processing. Structurally, it is executed before parsing takes place with the result of the tagging process being forwarded as an input to a syntax analyzer. Tagging is applied on a per-word level; the result of tagging is a tag which contains information regarding the word's Part of Speech. The Serbian language distinguishes between ten different parts of speech: nouns, verbs, numbers, adjectives, pronouns, conjunctions, prepositions, particles, adverbs, and interjections. POS tagging is implemented using classification. Classification is a supervised machine learning method which needs a set of data for training. In natural language processing this set of data is called a corpus of words. Those words are pre-tagged and used for training. Classification does not consist of only training, but it also requires an evaluation using the test set.

Syntax analysis for the Serbian language cannot be performed successfully by using only part of speech. Seeing as how many words of the Serbian language take many forms depending on their role in the sentence, a syntax analyzer needs more information in order to assign a syntax structure to a sentence. Some examples of the extra information needed are case for nouns, adjectives, pronouns, and numbers, whether a verb is a main or auxiliary verb, the type of pronoun, etc. There are not many corpora for the Serbian language. The one used for classification is srWaC [8]. srWac is a web corpus formed from .rs top-level domain. The first version contains 894 million of annotated tokens, and later versions around 600 million.

This corpus was chosen because it contains deep tags with lots of information. That being said, the corpus also contained too much information for some of the POS, so those values were filtered out in order to improve and simplify classification. After filtering out certain fields, an additional step of processing is applied in order to improve the usability of the corpus. The tags for pronouns contained information on pronoun type, whether it is personal, possessive, interrogative, indefinite, etc. This pronoun type is not as useful for syntax analysis as whether the pronoun is noun related or adjective related. This division impacts whether the pronoun can stand on its own or depends on some other word. Because of pronouns, srWaC corpus needed to be altered to only differentiate pronouns as noun or adjective related. These alterations are based on current type and some common word parts. For example, personal pronouns can only be noun related, demonstrative pronouns can only be adjective related, noun related pronouns often contain 'ko' or 'šta' inside the pronoun itself. The current pronoun tags were removed, and these rules are used to generate new tags for pronouns. After updating this corpus, it was necessary to train the classification model.

Classification is implemented using the nltk tool, more specifically nltk's Naive Bayes Classifier. First, the training was implemented without reducing some of the unnecessary information in tags, and this classification resulted in 89% accuracy. After tag reducing, accuracy rose to 90%. The final change in the corpus with pronoun tag simplification resulted in 91% accuracy.

IV. SYNTAX ANALYSIS

The nltk implementation of syntax analyzer uses a Context-free grammar of the target language. The definition of the grammar includes [3]:

- A starting symbol of the grammar,
- Terminal symbols – strings which contain tags that tagger recognizes,
- Grammar rules that have only one non-terminal symbol on the left side.

Nltk expects that the file which contains grammar rules also contains all the words that can be recognized in the target language as terminal symbols. The problem with this approach is that putting all words of one language in single file is nearly impossible, so the terminals of this grammar are not words, but rather tags that the POS tagger returned.

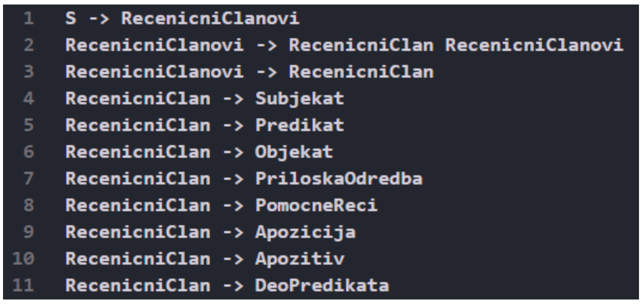
The CYK parser, on the other hand, uses a Context-free grammar in Chomsky Normal Form [9]. This means that every non-terminal can only be reduced as one terminal or two non-terminals. For rules

$$A \rightarrow B, \quad (1)$$

where A and B are non-terminals the parsing algorithm needs to be extended.

A. Grammar

The designed grammar is intended for both the nltk and CYK parser, so it is chosen to be in Chomsky normal form. The grammar covers all the syntax units that can be detected in one sentence of the Serbian language. The first part of the grammar covers the sentence structure. One sentence consists of sentence members, and sentence members can be subject, object, predicate, adverbial provision, apposition, etc. [10] Part of grammar that describes the structure of the sentence is shown in Fig. 1.



```
S -> RecenicniClanovi
RecenicniClanovi -> RecenicniClan RecenicniClanovi
RecenicniClanovi -> RecenicniClan
RecenicniClan -> Subjekat
RecenicniClan -> Predikat
RecenicniClan -> Objekat
RecenicniClan -> PriloskaOdredba
RecenicniClan -> PomocneReci
RecenicniClan -> Apozicija
RecenicniClan -> Apozitiv
RecenicniClan -> DeoPredikata
```

Fig. 1. Part of the grammar rules

For each of the sentence members a new set of rules is defined to describe them:

- Subject – there are two different types of subjects in the Serbian language. One is logical and the other one is grammatical. The logical subject is most likely to be in genitive, dative or accusative. The grammatical subject is a performer of the action expressed by predicate.
- Predicate – can be a noun or verb related. A verb predicate can consist of a main verb, an auxiliary verb, affirmative and negative particles, a verb in passive, etc. The set of rules for a verb predicate considers a list that consists of some of those parts. The noun predicate consists of a verbal part and a noun part. A verbal part is an auxiliary verb and a noun part can be a noun, adjective, or adverb.
- Object – can be direct or indirect. An object is an addition to the predicate. The main difference between direct and indirect objects is the case and the preposition absence.
- Adverbial provision – has similar definition as the indirect object and also serves as an addition to the predicate. The difference between the indirect object and the adverbial preposition is meaning in the sentence. The indirect object is a supplement to predicate, while the adverbial provision is more focused on giving information related to the action that predicate designates.
- Apposition and appositive - represent a syntax structure that is surrounded by commas and gives an alternative meaning to a noun or adjective it follows, respectively.
- Helper words – the words that have no syntactic meaning but are often part of the sentence and they must be covered by the grammar.

The total number of rules in the grammar is 406 so not all of the rules can be displayed. An example of a syntax tree built by defined grammar is shown in Fig 2.

While testing the NLTK parser with different texts, it has been noticed that some of the sentences were not recognized, which resulted in adding more rules in order to cover more sentence structures. A new problem that arose is the slowness of the NLTK parser, which led to a parsing duration of half an hour for a single sentence. Adding or altering the grammar rules became nearly impossible because the testing for 20-30 sentences lasted almost all day. The new parser using CYK algorithm was created as the solution for this problem.

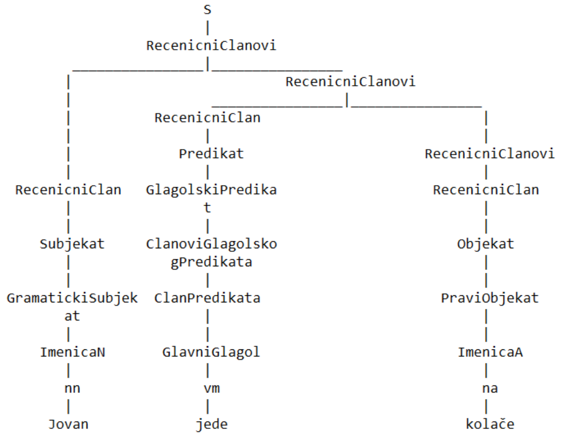


Fig. 2. Example of a syntax tree

B. CYK Parser

The CYK algorithm is a parsing algorithm for context free grammars, which uses a bottom-up search strategy. It uses a dynamic programming algorithm to tell whether a string is in the language of a grammar. There is a difference between the recognition and parsing algorithm. The recognition algorithm only shows whether a sentence is recognized by the grammar or not. The parsing algorithm returns all the possible syntax trees for a given input. The pseudocode that is used for CYK parser implementation is shown in Fig. 3.

```
function CYK-PARSE(words, grammar) returns table

  for $j$ ← from 1 to LENGTH(words) do
      $table[j-1, j] \leftarrow \{ A \mid A \rightarrow words[j] \in grammar \}$
      for $i$ ← from $j-2$ downto 0 do
          for k ← $i+1$ to $j-1$ do
              $table[i,j] \leftarrow table[i,j] \cup$
                  $\{ A \mid A \rightarrow BC \in grammar,$
                    $B \in table[i,k]$
                    $C \in table[k,j] \}$
              $table[i,j] \leftarrow table[i,j] \cup$
                  $\{ D \mid D \rightarrow A \in grammar \}$
```

Fig. 3. CYK parse pseudocode [4]

The CYK parser was implemented using Python. The parser implementation requires generating a table of dimensions (n+1) * (n+1) where n is the length of an input sentence. The idea is to fill out the table's diagonal with every possible production that has the current input as a terminal symbol in the grammar. After that, the parsing phase implements matching certain cells according to algorithm in order to climb up the tree until reaching starting symbol. Every cell is designed to memorize a list of nodes. The original pseudocode for CYK algorithm is expanded to fit the unary rules. Every time a new node is added to a list of nodes in a cell with indices [*i*][*j*], it is necessary to check whether that node is on the right side of some unary rule. If so, then the node to the left side of that unary rule is added to the node list. After parsing is completed, all possible syntax trees are found in cells with indices [*0*][*n*]. Generating the syntax trees consists of finding all the nodes that start with the starting symbol S and searching all the syntax trees by references.

V. Post-processing

The grammar is designed to recognize as many sentences as possible, but that also led to generating a lot of syntax trees for a single sentence. The NLTK parser and the CYK parser always generated the same trees, but that number for the longer sentences was often very large. In order to solve ambiguity, a new layer is added after parsing and generating the syntax trees and that layer is called the post-processing. The post-processing was implemented by eliminating trees that cannot be possible in the Serbian language. The rules that generate these kinds of trees couldn’t be left out from the grammar, so the solution was to search all the trees and to rule out the impossible ones. Some examples of post-processing rules are eliminating sentences that had multiple subjects and multiple predicates (every simple sentence has only one predicate). Also, the post-processing considered checking whether recognized apposition really has a noun in the same case on the left or the right side and whether recognized appositive has an adjective in the same case on the left or the right side, etc.

VI. Experimental results

The results of the implemented parser are shown with a comparison of how much time was needed for both parsers to generate the syntax trees for the same sentence. Besides the performance of the parser, it is tested how post-processing affected the reducing of the syntax trees.

The results for twenty sentences are shown in Table 1. The results show that parser implemented using CYK algorithm is thousands of times faster than the parser already implemented in the NLTK tool. Also, in most of the cases the post processing eliminated more than 50% of the trees and improved parsing results.

VII. Conclusion

In this paper, it is explained how the corpus for the Serbian language is updated and how the POS tagger for the Serbian language that uses updated corpus is created. It is described how the context-free grammar in Chomsky Normal Form for the Serbian language is created. Finally, it is presented creating of the syntax parser in two different ways using the previously mentioned context-free grammar. Based on the test results the CYK parser was 6115 times faster than the NLTK on average. Also, the post-processing layer after parsing reduced the number of syntax trees by 54% on average.

From all of the above, it can be concluded that although creating a new syntax analyzer based on the CYK algorithm and applying post-processing improved parsing and generated mostly three or less syntax trees there is still room for improvement. Improvement can be achieved by eliminating ambiguity entirely and creating a syntax parser that will only generate one syntax tree per sentence. One way to do that is by using Probabilistic Context-free grammars.

TABLE I

PARSER COMPARISON

Len	CYK trees	PP	PP imp. (%)	CYK time (s)	NLTK time (s)	NLTK / CYK (time)
6	2	1	50%	0.07	176	2514
10	8	2	75%	0.29	957	3300
7	2	1	50%	0.10	305	3050
7	26	9	65%	0.24	2301	9587
3	4	1	75%	0.01	224	22400
5	4	2	50%	0.07	350	5000
12	16	2	87%	0.76	1841	2422
10	52	9	83%	0.72	4975	6909
7	2	2	0%	0.12	179	1492
10	30	3	90%	0.48	2910	6062
6	6	6	0%	0.08	787	9837
4	1	1	0%	0.02	123	6150
5	2	1	50%	0.05	203	4060
8	10	2	80%	0.20	944	4720
5	10	2	80%	0.09	583	6477
5	4	2	50%	0.06	464	7733
6	2	1	50%	0.08	450	5625
8	18	6	66%	0.30	1813	6043
6	2	2	0%	0.08	212	2650
7	20	3	85%	0.21	1318	6276
Average:			54.3%			6115.3

Acknowledgment

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

References

[1] S. Bird, E. Klein, E. Loper, *Natural Language Processing with Python*, Sebastopol, USA, O'Reilly Media, 2009.

[2] D.H. Younger, “Recognition and parsing of context-free languages in n^3”, *Information and Control*, vol. 10, no. 2, pp. 189-208, 1967.

[3] M. Stanković, S. Stojković, Ž. Tošić, *Programski prevodioci*, Niš, Serbia, Elektronski fakultet, 2018.

[4] Jurafsky, J. Martin, *Speech and Language Processing, 2nd edition,* New York, USA, Prentice-Hall, 2009.

[5] J. Earley, “An Efficient Context-Free Parsing Algorithm”, Ph.D. dissertation, Carnegie-Mellon University, Pittsburgh, USA, 1968.

[6] R. M. Kaplan, *Natural Language Processing*, New York, Algorithmics Press, 1973.

[7] T. L. Booth, "Probabilistic representation of formal languages", 10th Annual Symposium on SWAT, pp. 74-81, Waterloo, Canada, 1969.

[8] srWaC – Serbian web corpus, http://nlp.ffzg.hr/resources/corpora/srwac/, 12.07.2020.

[9] N. Chomsky, “On certain formal properties of grammar”, *Information and Control*, vol. 2, no 2, pp. 137-167, 1959.

[10] Ž. Stanojčić, Lj. Popović, *Gramatika srpskog jezika za gimnazije i srednje škole*, Belgrade, Serbia, Zavod za udžbenike, 2011

Meme App Generator

Vladislav Tokovic, Luka Gajic, Dusan Cvetnic, Miodrag Vidacic, Nikola Savanovic, Miodrag Zivkovic

Abstract—**Memes are today part of cultural evolution, as a way of passing the information in form of a combination of image and text. Memes are all around us, being shared millions of times via social networks, different media sites or chat applications. In the work presented in this paper, we have aimed to develop application/service which would allow users to quickly access available meme templates, option to select any of them, add original top and bottom text (which is somewhat classic meme format) and quickly get their meme in the image format.**

***Index Terms*—memes, mobile application, Android, web service.**

I. Introduction

In its original sense, the word "meme" is a term for self-contained piece of cultural information that is transmitted by imitation [1]. The word originated from Greek "mimema", which translates to "imitated". The first mention of the word was in 1976., in the book on evolution "The Selfish Gene", by Richard Dawkins. The intention was to draw the analogy between biological and cultural evolution – same way the biological evolution depends on the genes, the units of information that are passed through generations, Dawkins considered cultural evolution to be function of similar entities. From that point of view, meme can be thought of as culture-carrying unit that is multiplied and transferred among individuals, during which it can experience random mutation and go through natural selection. This approach led to the field of study of culture and information called "memetics".

In this frame of reference, memes have several features: structure format can be a phrase, behavior, style; meme transmission format can be verbal, auditory, visual; meme transmission platform can be face-to-face, internet, newspaper. Richard Dawkins did not want to be associated with memetics, and that field of study has faced substantial criticism from scholars, which frequently characterized it as a pseudoscience. In this context, the concept of meme remains mostly theoretical.

However, what is undoubtedly visible, and present everywhere around us, is different type of memes – "internet memes" that have emerged around 2010 [2]. The term "internet meme" describes a different entity than the word "meme", as it stands for the idea or concept, in the form of image, gif, text, etc., that is substantially shared via social media sites or chat applications [3].

There is one fundamental distinction between meme and internet meme – meme experiences random mutations while internet meme can be intentionally modified by humans, and that is why these two terms are not interchangeable in the scientific sense [1].

Today people over the World use memes for learning math, history, any kind of literature or any scientific research field. For new generations of students is sometimes easier to remember formula, important sentence or definition of scientist, when it is written over the *Meme* [13]. The main goal of Meme generator is education and humor. Creation of funny and creative images is the simplest way that person can show his or her feelings [13]. One of studies which is published in 1993 is about text messaging and telecommunications, when some companies used emoticons in text messaging [14].

We aimed to develop application/service which would allow users to quickly access available meme templates, option to select any of them, add original top and bottom text (which is somewhat classic meme format) and quickly get their meme in the image format. That is the base functionality of our application, which we plan to improve in multiple aspects, described in the "Future Work" section.

II. Background and Motivation

Thanks to the internet memes, the word "meme" has experienced resurgence during this period, and ever since the inception, the meme-trend has been on the rise, with no indication of slowing down, as shown in Fig. 1.

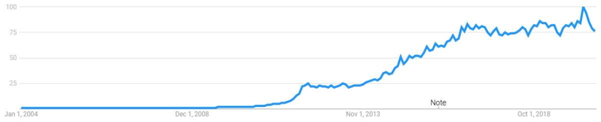


Fig. 1. Meme trends [4]

Internet memes have gained widespread popularity and

Vladislav Tokovic is with the Technical Faculty, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: vladislav.tokovic.19@singimail.rs).

Luka Gajic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: luka.gajic.17@singimail.rs).

Dusan Cvetnic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: dusan.cvetnic.17@singimail.rs).

Miodrag Vidacic, is with the Technical Faculty, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: miodrag.vidacic.19@singimail.rs).

Nikola Savanovic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: nsavanovic@singidunum.ac.rs).

Miodrag Zivkovic is with the Faculty of Informatics and Computing, Singidunum University, Danijelova 32, Belgrade, Serbia (e-mail: mzivkovic@singidunum.ac.rs).

mainstream recognition, following and acknowledgement but also a lot of criticism, especially when they are used (or being interpreted as if they are used) as a means of insult, degradation or discrimination, which escalated to the point of censorship on certain platforms [5].

This type of memes has made “reverse technological jump”, stemming from the virtual to physical world, with people expressing internet memes in person, wearing printed versions on their clothing, and even dressing up as internet memes.

Fig. 2. Person in the meme-inspired costume

The process of transmission of internet memes from one person to another can be realized through different ways of communications – via verbal, visual, audio, textual or electronic communication, also including the possible combination of mentioned means of communication. The platforms through which internet memes are distributed are also numerous – meme can be in virtually any format that can be perceived by at least one of the human senses, and in theory utilizing some sensory-signal enhancing or filtering tool such as microscope, telescope, or sound processing device.

Predominant platforms are social networking and social media websites and applications such as Reddit, Facebook, Twitter and 9GAG.

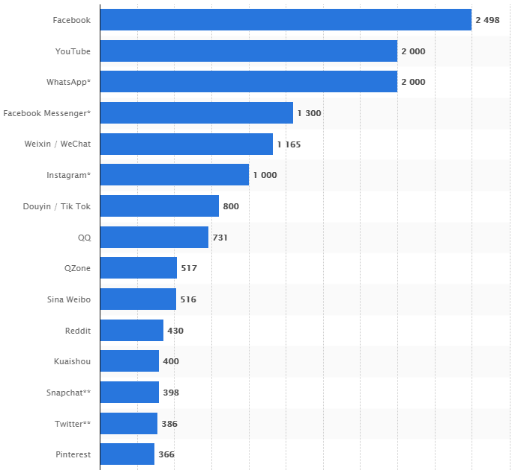


Fig. 3. Most popular social networks worldwide as of April 2020, ranked by number of active users, in millions [8]

Since majority of people refer to “internet memes” as just “memes” and to avoid overcomplicating terminology, from this point in the paper, we will use those terms interchangeably, both of them referring to what are by definition “internet memes”.

Given that internet memes have become one of the most notable and impactful formats of expression worldwide, we have concluded that there is a large market for innovative and easy-to-use meme creating application. Supporting that, is the fact that on Android Play Store there are seven meme creating applications with more than a million downloads, and out of those, three applications have more than 5 million downloads.

Here are several findings, from one social media behavior survey, that better reflect meme popularity [7]:

- 75% of 13 to 36-year-olds share memes
- 55% of 13 to 35-year-olds send memes every week
- 30% of 13 to 35-year-olds send memes every day
- 29% of 13 to 35-year-olds share memes on social media, 15% in chat applications and 31% in both places.

There are numerous recent publications dealing with the topics related to memes. For example, research presented in [9] shows a prototype tool that automatically generates memes from Portuguese news headlines. Socio-linguistic features of Internet memes in Macedonian has been assessed in [10]. The conclusions of this research suggest that numerous features make memes a very complex, but, on the other hand, very useful and creative internet communication tool. Having analysed over 100 top-rated memes from humour-dedicated web sites popular among Estonian users, in paper [11] it is discussed how humour creates intertextual references that rely partly on the cultural memory. Beside above-mentioned publications, there are numerous other publications, mostly dealing with the cultural and social aspect of memes, such as [12], and various software implementations for generating meme’s context.

III. Application Description and Back-end Implementation

In order to better present our idea, we have developed both android and web version of meme creating application. Key features of the android application are described below:

On application start, available meme templates are fetched and displayed on screen in the form of vertical list, as shown in Fig. 4.

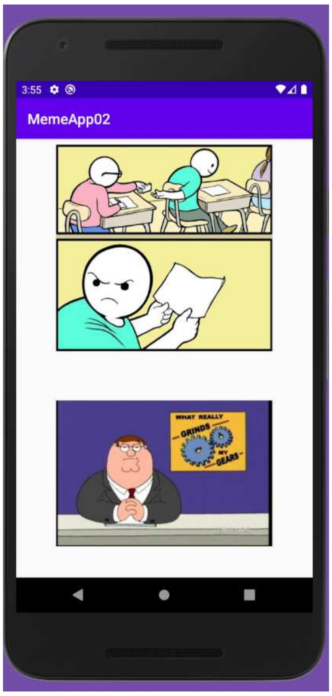


Fig. 4. Available templates when user starts application

User chooses the meme template by clicking the image on screen, once the image is selected, new window with chosen template is displayed. Below the image are labelled fields for top and bottom text input by the user, as shown in Fig 5.

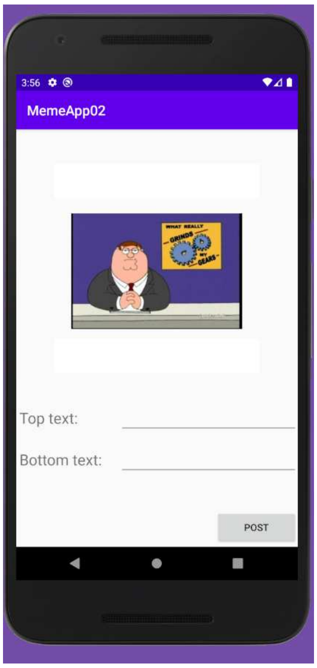
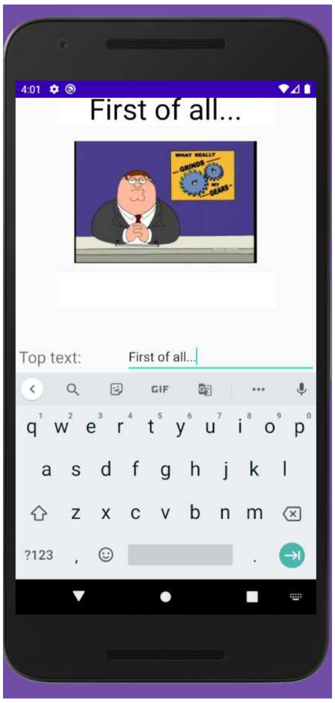


Fig. 5. Application ready for entering text

Simultaneously with the process of text input by the user, on every key press, that text is displayed on the appropriate place in front of the image, as shown in Fig. 6.



Top text: First of all...

Fig. 6. Displayed text on the top

In order to generate the meme, user should press the button bellow the input fields. When the button is pressed, on the back-end application side, the meme is created from the template and text input and displayed to the user in the in the image format, as shown in Fig. 7.

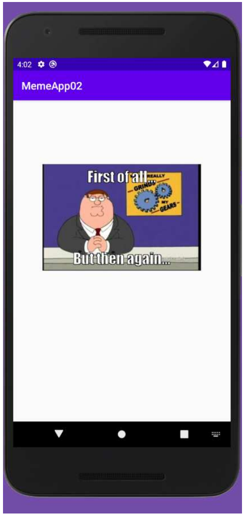


Fig. 7. Generated meme with custom text

Before starting the application back-end implementation, one of the main prerequisites is properly designed database. Since the application is still in the process of incremental development and more features and data will be provided after certain amount of time, currently chosen solution is database designed using relational model, with the help of MySQL RDBMS. The database itself is a store for two kinds of data:

- Meme templates – Previously added images, with normalized dimensions for giving better user experience, which users can choose, add their own text and generate new memes.

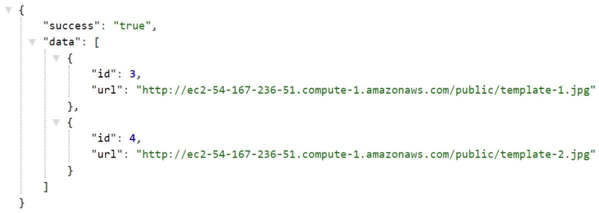
- Generated memes – User-generated pictures, related to their original image templates and proper URL for sharing on various social networks.

The authors used Node.js platform for the development of fast and event-driven modern Web applications using JavaScript programming language and V8 Google Chrome engine. Also, authors used Jimp (JavaScript Image Manipulation Program). Jimp is an image processing library for Node.js written entirely in JavaScript, with zero native dependencies. With Jimp, the text is added to the meme template, and created meme is stored in a database. The application is deployed using Amazon Web Services EC2 (Elastic Computing Cloud) service, since the application infrastructure required custom virtual machine configuration, instead of using one of available PaaS solutions.

Since it is possible that the application could gain more popularity during time, there is an option that it could migrate from relational to non-relational database solutions, such as MongoDB since they are able to deal with a large amount of data much better. The application layer is based on service-oriented architecture, more precisely it is based on REST (Representational State Transfer) which delivers data to multiple consumers using HTTP protocol, regardless of whether it is a Web browser or mobile application. After successful connection to database, the application creates multiple endpoints. The client has to read an API reference of the application so it can fetch and process JSON (JavaScript Object Notation) encoded data and also send new data in a proper way. For example, the HTTP POST request will be analyzed in order to present how can user send data to server and get back a generated image. User is obligated to provide 3 key-value pairs:

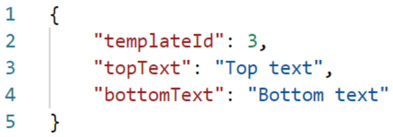
- Meme template ID
- Top text
- Bottom text

All of the request body parameters are mandatory and need to be JSON encoded beforehand. After sending data, the server returns new image ID and proper URL, as shown in Fig. 8, Fig. 9 and Fig. 10. respectively.



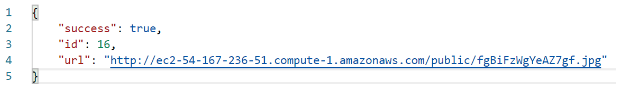
```
{
    "success": "true",
    "data": [
        {
            "id": 3,
            "url": "http://ec2-54-167-236-51.compute-1.amazonaws.com/public/template-1.jpg"
        },
        {
            "id": 4,
            "url": "http://ec2-54-167-236-51.compute-1.amazonaws.com/public/template-2.jpg"
        }
    ]
}
```

Fig. 8. Example of an endpoint for fetching all available meme templates



```
{
    "templateId": 3,
    "topText": "Top text",
    "bottomText": "Bottom text"
}
```

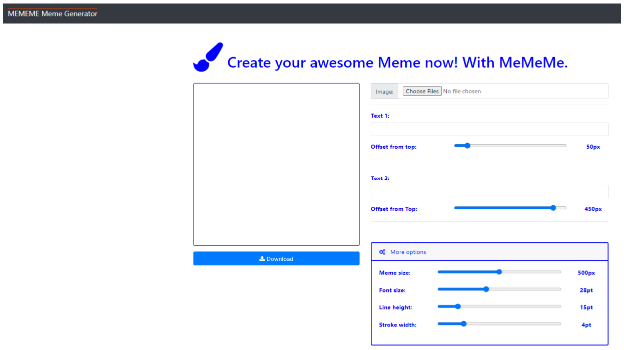
Fig. 9. Example of prepared POST request for creating a new meme



```
{
    "success": true,
    "id": 16,
    "url": "http://ec2-54-167-236-51.compute-1.amazonaws.com/public/fgBiFzWgYeAZ7gf.jpg"
}
```

Fig. 10. Example of returned response after sending a request for image creation

Browser version of our application is currently a stand-alone application and it does not request the template image from our API. Instead, it provides user with the option to choose images from their computer storage, as shown in Fig.11.



MEMEME Meme Generator

Create your awesome Meme now! With MeMeMe.

Image: Choose Files No file chosen

Text 1:

Offset from top: 50px

Text 2:

Offset from Top: 450px

More options

Meme size: 500px

Font size: 28pt

Line height: 15pt

Stroke width: 4pt

Fig. 11. Browser version of application

There are text input fields for bottom and top text, with the option, implemented using sliders, to change vertical text position on the image template, as shown in Fig. 12.

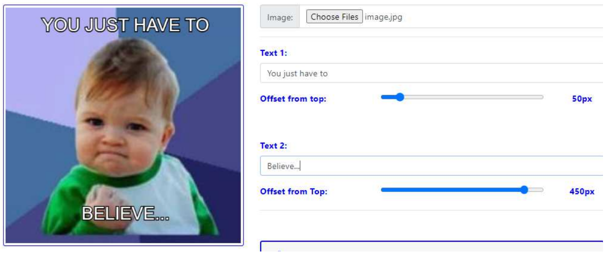


Image: Choose Files image.jpg

Text 1: You just have to

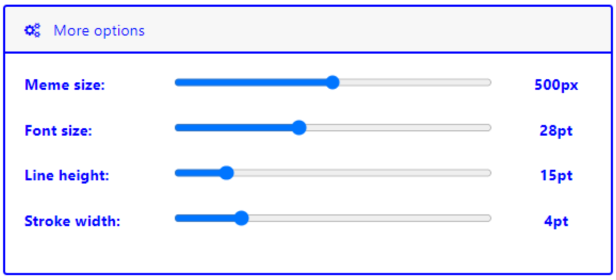
Offset from top: 50px

Text 2: Believe...

Offset from Top: 450px

Fig. 12. Text fields and custom options

Other editing options that are provided in the browser application are options to modify meme image size (in pixels), font size of the user-entered text (in points), line height (in points), stroke width (in points), as shown in Fig. 13.



More options

Meme size: 500px

Font size: 28pt

Line height: 15pt

Stroke width: 4pt

Fig. 13. Figure shows available options

When the user is finished with meme creation, they should click "Download" button in order to download the meme in the image format, to their computer, as shown in Fig. 14.

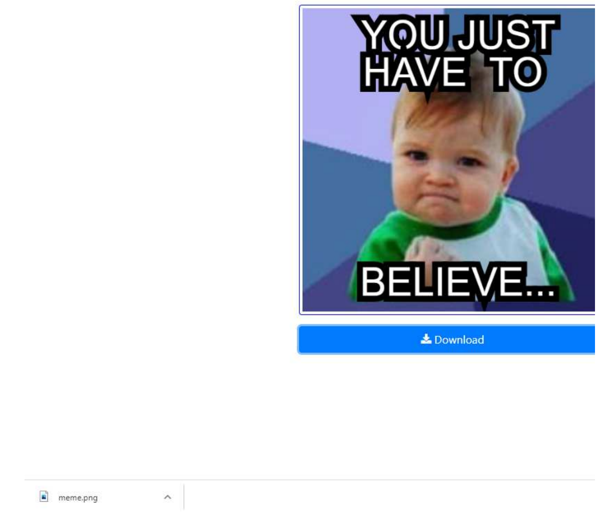


Fig. 14. Created image is downloaded

IV. Conclusion And Future Work

With the goal of application having success on the app market, we aim to improve the existing application functionality and user experience in several aspects:

1. Adding more features for modifying the selected meme template so that it has functionality of a basic image editor – crop, rotate, draw, change text fonts, combine images and similar. The distinction from the classic image editing application will be custom tailoring the service to meme-creating needs of the user.

2. Adding the option for user to access the camera through mobile application and take the photo for the meme in real-time.

3. Adding the option for making the gif and video memes.

The other major application part we plan to develop, which would distinguish this service from most of the popular meme-generating application, is adding the social network part, where the only content will be memes.

References

[1] https://www.britannica.com/topic/meme
[2] https://en.wikipedia.org/wiki/Memetics.
[3] https://www.wix.com/blog/2017/07/what-are-memes.
[4] https://trends.google.com/trends/explore?date=all&q=meme.
[5] https://reason.com/2019/03/11/learn-to-code-twitter-harassment-ross.
[6] https://www.facebook.com/HITZdotmy/posts/meme-cosplay-is-the-best-/10156135580893027.
[7] https://www.ypulse.com/article/2019/03/05/3-stats-that-show-what-memes-mean-to-gen-z-millennials/#:~:text=55%25%20of%2013%2D35%2D,sending%20them%20every%20single%20day.
[8] https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users
[9] Gonçalo Oliveira H., Costa D., Pinto A.M. "Automatic Generation of Internet Memes from Portuguese News Headlines". In: Silva J., Ribeiro R., Quaresma P., Adami A., Branco A. (eds) Computational Processing of the Portuguese Language. PROPOR 2016. Lecture Notes in Computer Science, vol 9727. Springer. 2016.
[10] Kostadinovska-Stojchevska, Bisera; Shalevska, Elena. "Internet Memes and their Socio-Linguistic Features". European Journal of Literature, Language and Linguistics Studies, [S.l.], v. 2, n. 4, nov. 2018.
[11] Laineste, L., & Voolaid, P." Laughing across borders: Intertextuality of internet memes". The European Journal of Humour Research, 4(4), 26-49, 2017.
[12] Marcus, O. R., & Singer, M. "Loving Ebola-chan: Internet memes in an epidemic". Media, Culture & Society, 39(3), 341-356, 2017.
[13] Linda K. Börzsei, Makes a Meme Instead: A Concise History of Internet Memes, New Media Studies Magazine (2013), Utrecht University
[14] Seth Godin , The Smiley dictionary, Berkeley, Calif. : Peachpit Press

Detection of Linux Malware Using System Tracers – An Overview of Solutions

Igor Vurdelja, Ivan Blažić, Dražen Drašković, Boško Nikolić

Abstract— **Linux is widely used for servers and embedded systems which require a high level of security and reliability. Although Linux is secure in general, traditional defense methods, such as signature-based detection, fail to detect new malicious programs. A more advanced approach is based on prediction of malicious behavior with dynamic analysis of the executed process. One method of observing the process execution on a Linux system is the use of system tracers such as ftrace and strace. This paper presents an overview of solutions for malware detection by using system tracers on Linux. Different malware detection strategies are discussed and compared with the presented approach. Results of several research projects done in this area are discussed, as well as the observed drawbacks. Technical details of this approach including the tracing utilities on Linux, sandboxing methods and machine learning models are discussed.**

Index Terms—**Computer Security, Dynamic Analysis, System Tracers, Linux.**

I. Introduction

Computer security is an important topic for many existing infrastructures. With the introduction of Internet of things (IoT), increasing number of devices have network access. These trends may bring additional convenience to everyday life, but also introduce new risks to security and privacy. Different threats can be identified for a particular system, such as DDoS attacks, sniffing attack, SQL injection, XSS, or even social engineering methods. Most of the currently used defense mechanisms are based on anti-virus tools, encryption, authentication, policies, user education, backups, or physical security [1].

Traditional defense mechanisms fail to detect zero-day malware as they are mostly based on identifying programs that are already known to be malicious [2][3]. A more complete defense includes protection against new malware. Accurate detection of new malicious programs is a difficult task due to the sophisticated behavior of malware [4] that hides its malicious functionality, and the number of false positives. This is a classification task that cannot be solved with the same methods as detecting existing malware.

One approach to this problem is the application of machine learning classification algorithms. In this approach a dataset with benign and malicious execution information is used to train a model that detects one or the other type of behavior [5][6][7][8]. System tracers present a universal solution for obtaining the execution information.

Linux is a popular choice for servers [9] and embedded systems [10], because of its benefits in performance, reliability and ease of development. These types of systems are at a higher security risk, as they may be a database server, network equipment, or a control unit of a safety critical device. The above-mentioned approach to detecting zero-day malware is applicable to Linux systems, as there are several system tracers available for this platform.

This paper presents an overview of research focused on malware detection using system tracers on Linux. The following section presents related research projects. Section III presents existing types of malware and detection methods. Section IV presents sandboxing environments for simulating malware execution, and section V presents the available system tracing tools on Linux. Pros and cons of the solutions described in the related research are discussed in section VI. Final section presents the conclusion of this paper.

II. Overview of Related Research

This section contains an overview of several researches of malware detection methods using system traces on Linux. Results of these researches, as well as the applied approaches are presented in this section and further discussed in the section VI.

Authors in [5] proposed an approach based on machine learning technique which uses system calls as features. Malware dataset is obtained from VX-heavens, and it contains 226 malware samples. Used dataset contains 226 malware and 442 benign samples. Results of the experiments presented in this research show a 97% classification accuracy. Malicious samples were executed in a virtual machine in order to collect system calls using strace. After each execution, the virtual machine was restored to clean state. System calls are divided in four categories: union, intersection, discriminating features for malware and discriminating features for benign programs. From each category the features are selected using multiple methods: Class Discrimination Measure (CDM), Odds Ratio (OR), Elimination of Sparse features (ESF). Authors examined how the choice of input data, classifier and feature length would affect the detection accuracy. In this study multiple classifiers are used: Naïve Bayes, J48, AdaboostMl (148), IBKS and Random Forest. It is experimentally shown that best accuracy is achieved with union and intersection sets as input data, Odds Ratio as feature selection method and Random Forest for classification. Accuracy of this model is 97,3%. It is also shown that discriminating features does not provide a good accuracy for distinguishing

Igor Vurdelja is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: vi195024p@student.etf.bg.ac.rs).

Ivan Blažić is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: blazic.ivan@outlook.com).

Drazen Draskovic is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: drazen.draskovic@etf.bg.ac.rs).

Bosko Nikolic is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: bosko.nikolic@etf.bg.ac.rs).

malwares from benign programs.

Research [6] was done by same authors as [5] and it is an upgraded version of technique described in the previous work. Authors presented 99,4% accuracy which is even higher than in the previous research. Accuracy was improved by changing the feature selection method. Previously used CDM, OR and ESF have been replaced with eXtended-Symmetric Uncertainty. In X-Symmetric Uncertainty, two-step dimensionality reduction is employed, and the features are ranked based on feature to class correlation and feature to feature inter-correlation. This approach allowed to pick features that had the biggest ability to predict the correct class and to eliminate features that are highly correlated, because they do not bring any useful information about the predicted classes. Best results have been achieved using Random Forest classifier with 27 features selected by eXtended-Symmetric Uncertainty. In this way, the authors managed to make a faster classifier with higher precision compared to their previous work.

Home routers and IoT devices are sensitive to DDoS malware. These devices usually run on Linux. One research [7] examined the application of system call analysis in order to classify behavioral anomalies occurring when the device is infected. Virtual machines running ARM based router firmware were used to simulate the victim device. Traffic simulation was done with a dataset of real anonymized internet traffic and ftrace was used to collect system calls. Two of the most popular DDoS botnet malware types for IoT device were detected, MrBlack and Mirai. These types of malware work by scanning the network for unsecured devices and infecting them by SSH or HTTP authentication with most commonly used credentials. After the device has been infected, it performs DDoS attacks and keeps spreading the malware in the same way. Raw system calls were processed for classification algorithms by extracting *n*-grams of system calls and applying TF-IDF transformation. This is an NLP approach that is commonly used to transform system call logs into machine learning data. Classification was done in different approaches, PCA anomaly detection, one-class SVM classification, naive detection based on identifying the set of *n*-grams that appears in normal traffic. Results of this research presented a 100% accuracy of all three approaches with sufficiently long *n*-grams. The rationale behind such a high accuracy of anomaly detection is the fact that these devices are highly specialized, therefore have strong patterns of normal behavior that are easily broken by malware.

A different approach was taken in a research that represents system calls as *hyper-grams* [8]. In this research the limitations of using *n*-grams for this kind of analysis is mentioned. These limitations mostly come down to observing system calls as raw sequences, without analyzing their individual functionalities and feeding the machine learning algorithms with small raw data that holds no generalized information about the characteristics of process behavior. *Hyper-grams* used in this research are a concise representation of system calls that occurred during a certain period of execution. Every system call is represented in a separate dimension, where the value in each dimension tracks the history of occurrences for that system call. This *hyper-gram* is parametrized with diminishing, addition, and sloping factors, that determinate the importance of new vs old information about specific system calls. These parameters allow a *hyper-gram* to contain a longer or a shorter history of a specific system call occurrence during the execution. For achieving best performance, genetic algorithms were used to optimize these parameters. In this way, a certain period of execution is described as a point in a multidimensional space. Benign and malicious points are identified by the number of their occurrences in benign and malicious processes. In-execution classification proposed in this research is based on re-calculating the point of the observed process with every new system call, and when the average point of this process is dominantly seen in malicious examples, the process is classified as malicious. An experiment was performed to test the approach, where 72 malicious and 72 benign processes were observed on a Linux virtual machine. Results of this experiment presented an AUC of 87,85 with the application of *hyper-grams*. For comparison, the same experiment was done with *n*-grams with different values of *n* and different classification algorithms (JRIP, J48, Naïve Bayes, SVM, and Instance based learner). Maximum AUC of 87,30 was achieved with Naïve Bayes for a 6-gram.

III. Malware Detection Strategies

Malware detection methods can be separated into different groups based on detection methods and types of program analysis. Common types of malware and detection methods are presented in this section.

A. *Malware identification methods*

The signature-based approach [2] is the most popular and widely used by commercial antivirus software. Signature-based methods rely on patterns extracted from malicious software binary files. This approach has a small error rate, but it cannot deal with simple obfuscation. Behavior-based [3] malware detection techniques observe the behavior of a program to conclude whether it is malicious or not. This approach has a better result in detecting polymorphic malware than a signature-based approach, but has issues detecting packers. The heuristic approach [11] relies on data mining and machine learning techniques to learn the behavior of a malicious program. This is the most modern approach which is still under research. It has the potential to resolve the issues of signature-based and behavior-based methods.

B. *Type of analysis*

During static analysis [12] the executable is analyzed on a file structure bases without execution. As the file is not executed, this type of analysis is fast and simple to deploy. On the other hand, using only static properties of an executable file might not accurately distinguish between benign and malicious executable due to malware's ability to modify or hide the binary code in order to preserve the same malicious behavior. The following techniques are commonly applied to hide the malicious properties of a program:

1) *Metamorphism*

Metamorphic malware will change its code on every execution. This is achieved by replacing existing instructions with a similar one. Despite the permanent changes to code, each iteration of metamorphic malware functions the same way.

2) Polymorphism

Polymorphic malware is a type of malware that changes its shape as well as signatures. It has two parts, but one of them will remain the same with each iteration, so this type of malware is easier to detect.

3) Packing

Packed malware is a type of malware that has been modified using some compression or encryption algorithm. The original executable is compressed, and an unpacked stub is appended to it. When loaded, the unpacked stub will unpack the whole executable and start original malware.

C. Limitations of analysis

Unlike static analysis, dynamic analysis does not include inspection of the binary code, instead, a malicious program is executed in a controlled environment. During this execution, malware will make a trace, or it will make behavior patterns that can be used to detect malicious behavior. The main advantage of dynamic malware analysis is that it is reliable for detecting unknown, metamorphic, and polymorphic malware. The disadvantage of dynamic analysis is that it is neither fast nor safe, and suffers from incomplete code coverage because it monitors only a single execution path. Additional limitation of dynamic analysis is the ability of modern malware to detect a controlled environment.

As static analysis suffers from severe limitations, this paper focuses on dynamic malware analysis using machine learning techniques. In the following chapters, the most common techniques used in dynamic malware analysis are presented.

IV. Malware Simulation Infrastructure

Dynamic malware analysis poses several challenges. The execution of malicious software can damage the host device, or another device connected to the same network. To protect the host from getting infected, a technique called sandboxing is used. This technique is often used by malware analysts to conduct dynamic analysis of untrusted files. A good sandboxing environment fulfills the following properties:

A. Secure isolation

The sandbox is an environment completely isolated from live systems, but it should simulate a live system to ensure that malware will run in the same way as in a real environment. By executing malicious code in a sandbox malware analyst can observe any impact on the potential victim, such as system configuration changes, network calls, or file system changes.

B. Ability to revert the environment to clean state

In automated malware analysis it is important that the system is reverted to a clean state every time new malware is analyzed. If the system is not reverted to a clean state data collected in this way is unreliable as the executed test may compromise the execution of the next test.

C. Realistic simulation of execution environment

Hiding a sandboxed environment consists of hiding virtual environments, as well as any tool used for malware analysis. Authors in [13] have shown that 17% of malware has mechanisms to detect a virtualized environment. Authors in [14] revealed that around 5% of the 110,000 malware samples attempted to evade analysis. Malware has a mechanism to identify whether they are being executed on a real system or a fake one. They show malicious behavior only after some period of time, or after a specific user action. A sandbox should have a mechanism to mimic real user behavior to provoke malicious behavior.

D. Automated execution of multiple samples

Development of a malware analysis method usually consists of executing a big number of malicious program samples. Therefore, an efficient and automated way of extracting traces for each malware has to be applied. Automated execution environments have the ability to uncover artifacts about the malware in a fast manner. There are several open-source tools for isolating malware execution, but they do not meet the mentioned criteria of a good sandboxing environment. On the other hand, these tools can be used for orchestration of multiple virtual machines on which the malware can be executed, and in this way help in automation of extracting malware traces. In the sense of providing a highly controlled environment, sandboxes may be a specific example of virtualization. Usually, sandboxes are implemented using virtual machines and nested virtualization, but authors in [15] also proposed sandboxing using Docker and LXC containers.

Some of the well-known open-source sandbox systems are Cuckoo and Lemon. They are very similar, and both rely on nested virtualization. On the first level of virtualization a virtual machine controls multiple virtual machines on the second level of virtualization. Malware is executed on the nested virtual machines to provide security and isolation. All nested virtual machines are in a virtual network together with a virtual machine on the first level. Fig. 1. Sandbox architecture presents the architecture of the described sandboxing environments.

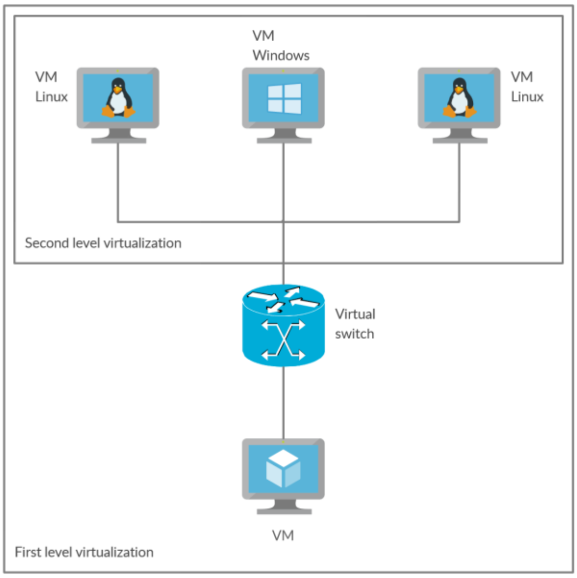


Fig. 1. Sandbox architecture

V. Tracing utilities on Linux

System tracers are used to obtain various information about the state of the system or executed processes. In the context of malware detection, data produced by tracers can be used to feed machine learning models that predict malicious behavior or anomalies. Authors in [16][17] showed that system calls can be used with high precision to recognize non-benign behavior.

Linux kernel has an extensive tracing infrastructure useful for debugging, and there are many tools available for system tracing on Linux. These tracers rely on the same mechanisms, but their use cases differ. A brief overview of kernel and user space tracers with tracing mechanisms is presented in Table 1 and Table 2. Most popular kernel tracers are presented in the following text.

TABLE 1
KERNEL TRACERS OVERVIEW

Tool Tracer	Function tracing	Static tracepoints	Trap	Trampoline
Ftrace	X	X	X	X
LTTng	X		X	X
Perf	X		X	X
eBPF	X		X	X
SystemTap			X	X

TABLE 2
USER SPACE TRACERS OVERVIEW

Tool Tracer	Function tracing	Static tracepoints	Trap	Trampoline
LTTng	X	X	X	
Printf		X		
Extrae	X			X

Ftrace is a tool that traces Linux kernel internal function calls and interactions between user space and kernel space. This tracer is included in Linux kernel since version 2.6.27. It is a framework of several sub-tracers from which the most typical is the function tracer. Ftrace can be used to trace kernel events like system calls, network traffic, memory access, etc. It relies on several tracing mechanisms including function instrumentation, static tracepoints, and dynamic tracepoints. Ftrace can be enabled or disabled in runtime. Ftrace is manipulated through a set of files in debugfs pseudo-filesystem. Aside from choosing a sub-tracer, ftrace configuration includes setting the size of the trace buffers, and selection of the clock source to use to timestamp the events. Users can choose which events to trace, so the execution overhead of ftrace is small when enabled, and negligible when disabled.

LTTng stands for *Linux Trace Toolkit: next generation*, and it is used for correlated tracing of the user applications, Linux kernel, and user libraries. LTTng dates from 2006, around the same time as ftrace, but LTTng is still not a part of the Linux kernel mainline. It consists of multiple Linux kernel modules for kernel tracing, and dynamically loaded libraries for user space tracing. There are multiple variations of LTTng besides the standard LTTng, these include LTTng-kprobe used for kernel tracing, LTTng UST and LTTng using tracefs for user space tracing. Within the same tracing session, the user can interact with multiple tracers. Most of today's development environments support a graphical interface to inspect LTTng logs.

Perf is a profiler tool for Linux 2.6+ based systems. It serves to monitor the performance of the system. Even though perf uses the same infrastructure as ftrace and LTTng, it is more commonly used as a profiler rather than a tracer. Perf has the ability not only to profile kernel functions but also user space applications. Perf can gather hardware PMU information such as different levels of cache misses, TLB misses, CPU cycles, missed branch predictions, etc. Perf on the other hand is limited to a single process. The events and counters reported by perf are those which occurred within the context of the traced process, and thus have been accounted for it.

SystemTap and eBPF allow users to write programs that can be inserted at run time at any location in kernel using Kprobes. They serve for aggregating and live monitoring of the system rather than for tracing, but can be re-configured to behave like tracers by writing probes that will make samples over time and store them.

Strace is a tool for system calls tracing. It does not come as a part of Linux kernel, so it must be additionally installed. It uses ptrace hooks infrastructure. Once attached to a process it will intercept all its system calls. Strace adds a large overhead, and is commonly used only for testing environments. LTTng and ftrace can achieve the same functionalities as Strace, but they are more lightweight.

VI. Analysis of the approach

This chapter presents an overview of the approach to detect malware on Linux using system traces based on existing research. Pros and cons of the methods applied in the previously presented research are discussed.

A. *Selection of the tracing tool*

Picking the suitable tracer is essential to get correct data and to make study expandable in the future. Most of the studies described in section II use strace to extract system calls from program samples.

Strace uses ptrace infrastructure, meaning that malicious file can easily check whether it is under analysis and can change its malicious behavior. Strace is not a standard part of Linux kernel and it must be additionally installed. This can be a pain point for embedded devices which use custom Linux build with Yocto or Buildroot. Strace is limited to tracing system calls only, so no additional information about system state can be obtained. This limits amount of information that could be used for a research. Due to lack of its portability and performance strace would not be suitable for a real-time malware detection tool.

Strace displays system call names and human readable arguments instead of system call numbers which allows an easier platform independent interpretation of system logs. It does not require root privileges and in terms of usability is by far the easiest tracer to be used on Linux. Arguments can be used to enhance the model to be more accurate in detecting malicious binary files. Mentioned limitations of strace can be overcome by using another tracer such as ftrace.

B. Sandbox environment

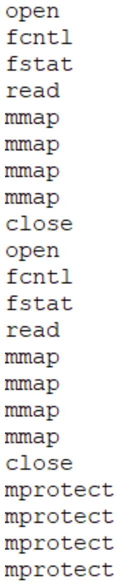
Properties of a good sandboxing environment are presented in section IV. Quality of the sandboxing infrastructure can greatly impact the results, but most of the presented papers lack detailed descriptions of these environments.

C. Machine learning applications

A common approach to the classification problem in the related research is machine learning. This kind of approach requires well prepared datasets and an accurate quality evaluation method to be reliable.

All research mentioned in section II suffer from imbalanced or incomplete datasets. Datasets are either small and contain up to 700 samples or are heavily imbalanced containing 80% of malwares and 20% of benign programs. *N*-grams of system calls are usually used as a feature set. When a dataset is small and has a vast number of features the problem of sparse data occurs. In case of *uni*-grams, number of features is around 350, and in case of *bi*-grams, number of features is around 350^2. In these cases classification can be improved by using one-class algorithms, reinforcement learning or obtaining more data.

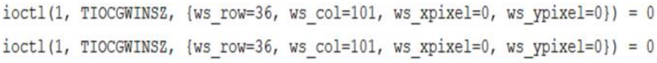
Malwares are usually distributed as statically linked executables for improved portability, which allows them to infect more devices. On the other hand, benign elf files are usually dynamically linked and taken from `/bin`, `/sbin`, `/usr/bin` or `/usr/sbin`. When executing dynamically linked executables system calls for loading shared libraries can occur. An example of a sequence for loading shared libraries is shown on Fig. 2. Important observation is that the majority of malware does not contain these sequences as they are statically linked, but benign programs do. This can introduce a bias towards classifying statically linked executables as malware, and dynamically linked as benign, which could explain a very high accuracy in previously presented research. Two ways of overcoming this problem are compiling Linux commands as statically linked executables or filtering sequences related to loading shared libraries.



```
open
fcntl
fstat
read
mmap
mmap
mmap
mmap
close
open
fcntl
fstat
read
mmap
mmap
mmap
mmap
close
mprotect
mprotect
mprotect
mprotect
```

Fig. 2. System call sequence for loading shared libraries

Linux commands are used as benign samples in most of the presented research, but criteria for choosing individual commands has not been described. Most of the Linux commands access the file system, but malicious samples usually make significantly more network calls, which can lead to inaccurately biased models. Vast majority of benign Linux commands print on the console while malicious files do not, which can lead to a similar bias problem. System calls used for obtaining information about the console is shown on Fig. 3.



```
ioctl(1, TIOCGWINSZ, {ws_row=36, ws_col=101, ws_xpixel=0, ws_ypixel=0}) = 0
ioctl(1, TIOCGWINSZ, {ws_row=36, ws_col=101, ws_xpixel=0, ws_ypixel=0}) = 0
```

Fig. 3. System calls for getting information about console

System call sequences made by both benign and malicious samples should be as diverse as possible in order to avoid this kind of bias. This can be achieved by a proper selection of benign samples.

An NLP approach has been mostly used, where system trace entries are treated as a set of individual smaller sequences which are then processed with *bag-of-words* algorithms. The NLP approach is considered to be robust in general but could lead to a less reliable classification if normalization such as TF-IDF is not applied or the size of *n*-grams is not optimal. In the context of malware detection, this approach can identify smaller patterns of execution that can frequently appear only in malicious programs. This approach lacks complete information about the execution since the individual sequences of a program are not independent. Data representation using feature vectors such as *hyper-grams* overcomes this limitations to a certain extent, but does not seem to provide significant improvement over *n*-grams [8]. High classification accuracy is presented in the related research, where most results show over 90%. This amount of accuracy is mathematically correct, but may not reflect the predictive power the classifier has in reality [18]. In cases where small or unbalanced datasets are used, a high accuracy will give a false measure of quality.

VII. Conclusion

Analysis of Linux system trace logs for malware classification has been a topic of several research presented in this paper. Approaches presented in the related research consist of executing benign and malware programs in a controlled environment to obtain system trace logs, which are then processed with machine learning algorithms to implement a classifier. Most results of these research present high classification accuracies but may lack appropriate datasets and real predictive power. From a critical analysis of the related research, it has been concluded that this approach requires a bigger, more well-prepared machine learning dataset, and a realistic simulation of the execution. Presented approach to malware detection has a lot of further research potential, with the ultimate goal of having robust real-time detection of zero-day malware on Linux systems.

REFERENCES

[1] Lee Brotherston, Amanda Berlin, *Defensive Security Handbook: Best Practices for Securing Infrastructure*, Newton, Massachusetts, US, O'Reilly Media, Inc, 2017.

[2] P. Gutmann. "The Commercial Malware Industry", available at https://www.cs.auckland.ac.nz/~pgut001/pubs/malware_biz.pdf

[3] W. Liu, P. Ren, K. Liu, H. Duan, "Behavior-Based Malware Analysis and Detection", First International Workshop on Complexity and Data Mining, NW Washington, DC, United States, pp. 39-42, 2011.

[4] Emanuele Cozzi, Mariano Graziano, Yanick Fratantonio, Davide Balzarotti, "Understanding Linux Malware", IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, pp. 161-175, 2018.

[5] K. A. Asmitha, P. Vinod, "A Machine Learning Approach for Linux Malware Detection", International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), Ghaziabad, India, pp. 825-830, 2014.

[6] K. A. Asmitha, P. Vinod, "Linux Malware Detection Using eXtended-Symmetric Uncertainty", International Conference on Security, Privacy, and Applied Cryptography Engineering, Pune, India, pp. 319-332, 2014.

[7] N. An, A. Duff, G. Naik, M. Faloutsos, S. Weber and S. Mancoridis, "Behavioral anomaly detection of malware on home routers," 12th International Conference on Malicious and Unwanted Software (MALWARE), Fajardo, Puerto Rico, United States of America, 2017.

[8] B. Mehdi, F. Ahmed, S. A. Khayyam, M. Farooq, "Towards a Theory of Generalizing System Call Representation for In-Execution Malware Detection", IEEE International Conference on Communications, Cape Town, South Africa, 2010.

[9] "Usage Statistics and Market Share of Operating Systems for Websites, August 2020", W3Techs, accessed 09.09.2020, https://w3techs.com/technologies/overview/operating_system.

[10] "2019 Embedded Markets Study", AspenCore, accessed 09.09.2020, https://www.embedded.com/wp-content/uploads/2019/11/EETimes_Embedded_2019_Embedded_Markets_Study.pdf.

[11] Z. Bazrafshan, H. Hashemi, S. Mehdi Hazrati Fard, A. Hamzeh, "A survey on heuristic malware detection techniques", 5th Conference on Information and Knowledge Technology, Shiraz, Iran, pp. 113-120, 2013.

[12] M. G. Schultz, E. Eskin, E. Zadok, S. J. Stolfo, "Data mining methods for detection of new malicious executables", Proceedings of IEEE Symposium on Security and Privacy (S&P), Oakland, California, pp 38-49, 2001.

[13] "Carbank Malware – Ninety Five Percent Exhibits Stealthy or Evasive Behaviors", available at https://www.lastline.com/labsblog/carbanak-malware-ninety-five-percent-exhibits-stealthy-or-evasive-behaviors/, date accessed 8. June 2020.

[14] D. Kirat, G. Vigna, C. Kruegel, "BareCloud: Bare-metal Analysis-based Evasive Malware Detection", USENIX Security Symposium, San Diego, California, pp. 287-301, 2014.

[15] D. Hellinger, L. M. Xuan, P. Gahlot, "Dynamic Analysis of Evasive Malware with a Linux Container Sandbox", available at https://www.researchgate.net/publication/330500642_Dynamic_Analysis_of_Evasive_Malware_with_a_Linux_Container_Sandbox

[16] S. Forrest, S. Hofmeyr, A. Somayaji, T. Longstaff, "A sense of self for Unix processes", IEEE Symposium on Security and Privacy, Oakland, California, pp. 120-128, 1996.

[17] K. Denney, C, Kaygusuz, J. Zuluaga, "A Survey of Malware Detection Using System Call Tracing Techniques", 2018.

[18] F. J Valverde-Albacete, C. Peláez-Moreno "100% Classification Accuracy Considered Harmful: The Normalized Information Transfer Factor Explains the Accuracy Paradox", PloS one, 2014

IoT-based System for COVID-19 Indoor Safety Monitoring

Nenad Petrović and Đorđe Kocić

***Abstract*—In this paper, we introduce an affordable IoT-based solution aiming to increase COVID-19 indoor safety, covering several relevant aspects: 1) contactless temperature sensing 2) mask detection 3) social distancing check. Contactless temperature sensing subsystem relies on Arduino Uno using infrared sensor or thermal camera, while mask detection and social distancing check are performed by leveraging computer vision techniques on camera-equipped Raspberry Pi.**

***Index Terms*—Arduino; computer vision; coronavirus; COVID-19; Raspberry Pi, ontology.**

I. Introduction

Since the last days of the previous year, the occurrence of novel infectious flu-alike respiratory disease *COVID-19* caused by *SARS-Cov-2* virus (also known as *coronavirus*) has affected almost every aspect of people's lives globally. First, it was discovered in China, but spread quickly to other continents in just few weeks. According to [1], until July 11th, 2020, the total number of identified cases was 12,653,451, while taking 563,517 lives worldwide.

Common symptoms of coronavirus disease include fever, tiredness, sore throat, nasal congestion [2], loss of taste and smell [3]. In most cases, it is transmitted directly (person to person) through respiratory droplets, but also indirectly via surfaces [4, 5]. Incubation period could be quite long and varies (between 14 and 27 days in extreme cases) [6, 7]. Furthermore, even asymptomatic persons (almost 45% of cases) can spread the disease [7] making the situation even worse. Therefore, the usage of face masks and sanitizers has shown positive results when it comes to disease spread reduction [8]. However, the crucial problem is the lack of approved vaccine and medication [9].

Due to these facts, many protection and safety measures were taken by governments in order to reduce the disease spread, such as obligatory indoor mask wearing, social distancing, quarantine, self-isolation, limiting citizens' movement within country boarders and abroad, often together with prohibition and cancellation of huge public events and gatherings [10]. Despite the fact that the pandemic seemed weaker at some points, most of safety regulations are still applied due to unstable situation. From workplace behavior to social relations, sport and entertainment, coronavirus disease poses many changes to our everyday routine, habits and activities.

In this paper, cost-effective IoT-based system aiming to help organizations respect the COVID-19 safety rules and guidelines in order to reduce the disease spread is presented. We focus on most common indoor measures - people with high body temperature should stay at home, wearing mask is obligatory and distance between persons should be at least 1.5-2 meters. For the first scenario, Arduino Uno microcontroller[1] board with contactless temperature sensor is used, while we rely on Raspberry Pi[2] single-board computer equipped with camera making use of computer vision techniques for other two scenarios. We decided to use these devices due to their small size and affordability.

II. Background

A. *OpenCV*

Python version of OpenCV [11], open-source computer vision library was used for implementation of mask detection and social distance check algorithms. We decide to use it, as it was approved for usage with older Raspberry Pi devices [12].

Face and body detection algorithms rely on the existing OpenCV implementation of Viola-Jones object detection framework based on Haar feature cascades [13]. It is a machine learning approach where cascade function is trained from a large set of positive and negative images. After that, this function is used to detect objects in new images. OpenCV comes with both trainer and detector. However, OpenCV offers pre-defined classifiers for detection of commonly used objects, such as human face, whole body, body and face parts (both front and back for some of them). Therefore, in this paper, we leverage the existing classifiers provided by OpenCV library, as they were enough to cover satisfy the needs of the implemented solution.

In [12], face detector provided by OpenCV library was used for control of multimedia reproduction systems based on Raspberry Pi devices within museums and cultural heritage sites, showing acceptable performance, even in real-time use cases.

B. *MQTT*

In this paper, MQTT (Message Queuing Telemetry Transport)[3] was used for machine-to-machine communication between the involved devices - Raspberry Pi, Arduino, Edge

Nenad Petrović is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: nenad.petrovic@elfak.ni.ac.rs).

Đorđe Kocić is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: seriousdjoka@gmail.com).

[1] https://store.arduino.cc/arduino-uno-rev3

[2] https://www.raspberrypi.org/

[3] http://mqtt.org/

servers and smartphones. It is a lightweight, publish-subscribe messaging protocol on top of TCP/IP. MQTT is designed for use cases where small code footprint is desired or network bandwidth is limited, which is suitable for IoT solution leveraging low-power computing devices presented in this paper. Moreover, publish-subscribe messaging mechanism requires a message broker. For that purpose, we use a Node.js MQTT broker implementation within Node-RED[4] deployed on a server residing within the Edge. For IoT devices, corresponding MQTT client libraries were used – PubSubClient[5] for Arduino, Paho-MQTT[6] for Raspberry Pi and Paho Android Service[7] for smartphones. The devices measuring body temperature, detecting masks and social distancing send MQTT messages to Edge servers in cases when a person does not satisfy conditions to pass some of the safety check steps. Furthermore, the Edge server processes the received messages and forwards them to corresponding security workers to notify them about breaking of COVID-19 safety rules. Each message is sent in a form of JSON-encoded string.

C. *Semantic knowledge representation*

The role of semantic technology is to enable encoding the meaning of data separately from the content itself and related applications, which provides the ability to understand data, exchange its understanding and perform reasoning on top of it [14]. In this case, the formalization of knowledge is done in a form that is understandable for both humans and machines. Within the semantic knowledge bases, the data is represented with respect to ontologies.

In IoT systems, ontologies are often used to achieve interoperability and integrate data coming from heterogeneous devices and their sensors in order to enable their control in unified way [15]. For example, in [16], we introduced Smart Grid Response Ontology to enable reasoning about the events that occurred within the IoT-based system for smart grid monitoring and generate adequate response in a particular context. In this paper, we adopt similar approach to semantic representation in context of COVID-19 safety monitoring, but extend it with the elements of spatial reasoning.

III. Related work

There are several existing works that contain some of the elements relevant to the work presented in this paper. However, to the best of our knowledge, there is no such solution covering all these aspects together to achieve this goal while allowing execution on low-cost IoT devices at the same time.

In [17], a dataset for masked face recognition is introduced and its application by different algorithms in context of campus and enterprise coronavirus prevention discussed. Moreover, in [18], a high-accuracy method for facial mask detection using semantic segmentation based on fully convolutional networks, gradient descent and binomial cross entropy was presented. However, performance-wise, it is too heavy for low-power IoT devices, such as Raspberry Pi. On the other side, in [19], a state model-based solution for face mask detection relying on Viola-Jones algorithm in context of ATM center security was described.

When it comes to temperature sensing, there are several variants of Arduino-based solutions. In [20], Arduino was used for real-time temperature visualization using MATLAB. However, the used sensor does not allow contactless temperature sensing. Moreover, in [21], a similar system incorporating the usage of smartphones for remote temperature monitoring using Arduino Uno was presented.

The system architecture presented in this paper is inspired by our previous work on remote smart grid monitoring anomaly, power consumption monitoring and relay protection using IoT devices (smartphones and Arduino Uno) [16] and video surveillance system relying on Raspberry Pi single-board computers and Edge servers [22]. Our main goal is to provide a comprehensive solution for COVID-19 safety monitoring which relies on IoT devices as much as possible in order to be affordable at the same time.

IV. Implementation

A. *System overview*

Our solution consists of the following subsystems: 1) temperature measurement subsystem based on Arduino Uno 2) computer vision subsystem for mask detection and social distancing check based on Raspberry Pi 3) server side 4) smartphone application for security guards.

First, all people that try to the enter building have to pass contactless temperature check. For that purpose, we rely on Arduino Uno equipped with infrared thermometer (such as MLX90614[8]) or thermal camera sensor (AMG8833[9] for example). Moreover, it uses ESP8266 WiFi module for communication with Edge servers using MQTT protocol. In case that person has body temperature higher than normal, the door is locked and MQTT message sent to server, containing both the temperature value and location where it was recorded. Server receives this message, parses it and forwards to smartphone application used by security guards, so they can arrive to make sure that person does not try to enter the building further. Otherwise, if passenger's temperature is normal, Arduino will send signal to open the door.

After that, passengers proceed to next step of checking – mask detection. For this task, computer vision subsystem based on Raspberry Pi single-board computer equipped with camera module version 1[10] revision 3 was used. In case that passenger does not wear mask or it does not cover nose, security guards will be informed via MQTT message, so they can provide a mask or warn that person to leave. Otherwise, if

[4] https://nodered.org/
[5] https://github.com/knolleary/pubsubclient
[6] https://pypi.org/project/paho-mqtt/
[7] https://github.com/eclipse/paho.mqtt.android

[8] https://maker.pro/arduino/projects/build-an-infrared-thermometer-arduino-and-mlx90614
[9] https://learn.adafruit.com/adafruit-amg8833-8x8-thermal-camera-sensor/arduino-wiring-test
[10] https://www.raspberrypi.org/documentation/hardware/camera/

the person that is being checked wears mask, the door will be opened. Furthermore, once they enter the building, Raspberry Pi devices check whether social distancing is applied properly or not at given locations. In a similar way, MQTT message will be sent to inform the security guards when social distancing is not applied properly in some of the rooms.

On the server side, the MQTT broker and semantic triple store are deployed, while message processing, event logging, reasoning and message forwarding are done. Edge servers receive messages, perform their semantic annotation and reasoning to find the right security guard that will be notified. A simple Android mobile application used by security guards receives MQTT messages from server side and visualizes the data about rule violation and location where it occurred within the building. In Fig. 1, an overview of the proposed IoT-based solution that aims to ensure that COVID-19 safety guidelines are applied properly indoors is given.

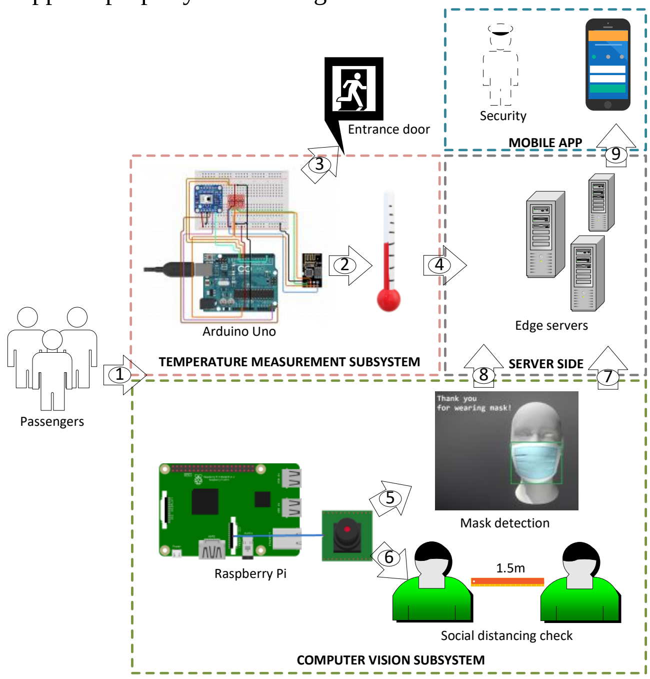


Fig. 1. COVID-19 indoor safety IoT system overview: 1-Passenger arrival 2-Temperature value 3-Door open/close signal 4-MQTT message warning that someone has higher body temperature than average 5-Wears mask/doesn't wear mask 6-Social distancing satisfied/not satisfied 7-MQTT warning message telling that person without mask tries to enter 8-MQTT warning message that passengers do not respect social distance measures 9-MQTT notification messages sent to security worker's smartphone about people breaking COVID-19 safety measures at various building rooms.

B. *Mask detection algorithm*

For implementation of mask detection algorithm, we rely on three OpenCV library classifiers[11]: 1) *haarcascade_frontalface_default* – which is used for detection of human face from frontal side 2) *haarcascade_mcs_mouth* – recognizes human mouth within the provided image 3) *haarcascade_mcs_nose* – to detect nose. For each frame coming from camera stream, the procedure given in Listing 1 is executed.

```
Input: image
Output: label
Steps:
gray_image = ConvertToGray(image);
bw_image = ConvertToBw(gray_image);
faces = DetectFaces(gray_image);
faces_bw = DetectFaces(bw_image);
if(faces.length==0 and faces_bw.length==0)
   label = "No face found!"
else
   if(faces_length==0 and faces_bw.length>0)
      mouths = DetectMouth(bw_image);
      noses = DetectNose(bw_image);
    else
      mouths = DetectMouth(gray_image);
      noses = DetectNose(gray_image);
   end if;
    if(mouths.length==0 && noses.length==0)
       label = "Thank you!"
       OpenDoor();
    else
        for each m in mouths
           for each f in faces
              if(m.y> y and m.y<f.y+f.h and m.x>x   and m.x<f.x+f.w)
                 label = "Please put mask on!";
                 sendMQTT("no mask", "location name");
              end if;
           end for;
        end for;
        for each n in noses
           for each f in faces
              if(n.y> y and n.y<f.y+f.h and n.x>x   and n.x<f.x+f.w)
                 label = "Please put mask over nose!";
                 sendMQTT("improper mask", "location name");
              end if;
           end for;
        end for;
    end if;
end if;
return label;
end.
```

Listing 1. Pseudo-code of mask detection algorithm.

First, the camera frame is converted to a gray scale image which is used for face detection, as it required by the OpenCV's Haar cascade classifier. Moreover, additional black and white version of camera frame is also created as a new copy. It was empirically noticed that in most cases of person wearing a white mask, OpenCV classifier cannot identify the face correctly if gray scale image is used, but performs better with black and white images instead.

After that, face detection is performed against both images. In case that array length of detected faces is 0 in both cases, then it is assumed that there isn't any human present within the camera's view. Otherwise, if a face was detected, mouth and nose detection are further applied to the corresponding camera frame version. In case that image does not contain mouth and nose, it means that person wears mask properly and corresponding door will be opened. However, if mouth is detected and its coordinates are within the area of the detected face, then the person is warned that mask is needed in order to proceed. If nose is detected within face area, then the person is warned to put the mask properly (covering nose). The algorithm is working in both single- or multi- person mode. In single-person mode, it is assumed that people pass one by one

[11] https://github.com/opencv/opencv/tree/master/data/haarcascades

near the mask detection system's camera. In case of multi-person mode, the check whether mouth and nose coordinates lie inside a face is done for each pair of face-mouth and face-nose pair, as more than one person can be allowed to pass near camera at once. In Fig. 2, a screenshot of successful mask detection on Raspberry Pi model 2B is given.

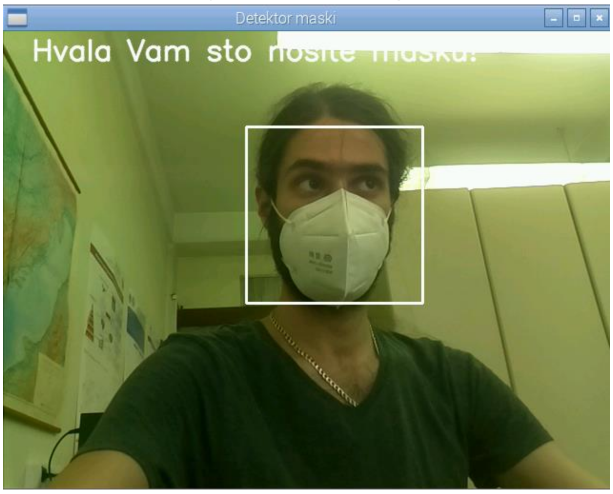


Fig. 2. Mask detection on Raspberry Pi 2B.

C. Social distancing check algorithm

When it comes to social distancing check algorithm, it leverages OpenCV's *haarcascade_fullbody* classifier for human body detection within the captured image. In Listing 2, the previously described algorithm for social distancing check based on computer vision is given.

```
Input: image, threshold_d
Output: label
Steps:
gray_image=ConvertToGray(image);
bodies=DetectFaces(gray_image);
if(bodies.length≤1)
    text="Not enough people for check!"
else
    for each b1 in bodies
        for each b2 in bodies
            d=sqrt((b1.x-b2.x)^2 +(b1.y-b2.y)^2);
            d_m=ConvertPixelsToMeters(d);
                if(b1 ≠ b2 and d_m< threshold_d)
                    label="Social distancing not applied!";
                    sendMQTT("social distancing alert", "location name");
                end if;
end if;
return label;
end.
```

Listing 2. Pseudo-code of social distancing check algorithm.

In a similar way as mask detection algorithm, each camera frame is converted to a gray scale image. Furthermore, body detection is applied. If more than one human body is detected, the distance between each two persons is calculated and compared against a given threshold distance *threshold*$_d$, given in meters. However, all distances should be normalized depending on the camera characteristics and object position before comparison. The mapping of pixels to real world distances in meters is performed with respect to formula [23]:

$$\frac{image\ dimension}{focal\ length} = \frac{object\ dimension}{distance\ to\ object} \tag{1}$$

If the distance between each two bodies is greater or equal to *threshold*$_d$, then social distancing is applied correctly in a given scenario. Otherwise, if this condition does not hold for at least one pair of bodies, then the message will be sent to the server and security operator notified. Fig. 3 shows the screenshot of social distance check application.

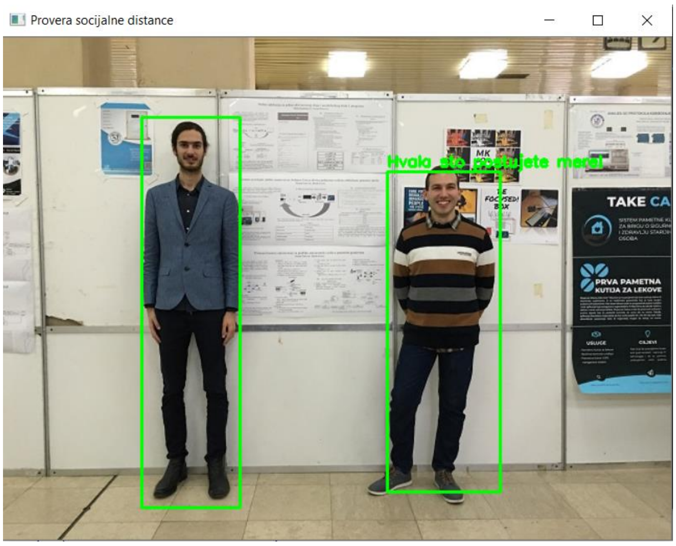


Fig. 3. The authors testing the social distancing check algorithm for threshold$_d$=1m.

D. Temperature measurement

In Listing 3, an excerpt of code running on Arduino is given.

```
void loop() {
    temp = sensor.readCelsius();
    if (temp > 37) {
            digitalWrite(13, HIGH);
            snprintf (msg, 100, "Temperature:#%ld;
location:faculty entrance", temp);
            client.publish("tempTopic", msg);
    }
    else {
            digitalWrite(13, LOW);
    }

}
```

Listing 3. Excerpt from temperature measurement code run on Arduino Uno positioned at faculty building entrance.

The temperature measurement subsystem based on Arduino Uno measures passenger's temperature using contactless IR sensor. The passengers pass one by one. In case that passenger's temperature exceeds average human body (37 °C), then Arduino Uno generates signal to lock the door in order to prevent the person from entering the building and sends MQTT message which tells that person with high body temperature was detected at a certain location. Otherwise, the door is opened to let the person in.

E. COVID-19 Indoor Safety Monitoring Ontology

The highest-level concept in this ontology is *Monitoring System*. It consists of heterogeneous *Devices*, such as Raspberry Pi, Arduino Uno and conventional laptop. Each device is equipped with different sensors and is able to detect different types of safety rule *Violations*. Several types of safety rule violations are considered: *Social Distancing*, *No Mask* and *High Temperature*. In case that some violation occurs, then the corresponding *Action* is taken as a response, such as closing *Door* or notifying *Security Guard*. For both the physical objects (*Device*, *Door*, *Security Guard*) and *Violation* events, the *Room* where it resides or occurs is relevant. Each *Room* is located on a *Floor*. This way, it is enabled to find the available *Security Guard* from the same *Floor* where *Violation* occurred and send him/her notification. Otherwise, the first guard that is available is selected. In Fig. 4, an excerpt from the described ontology is given.

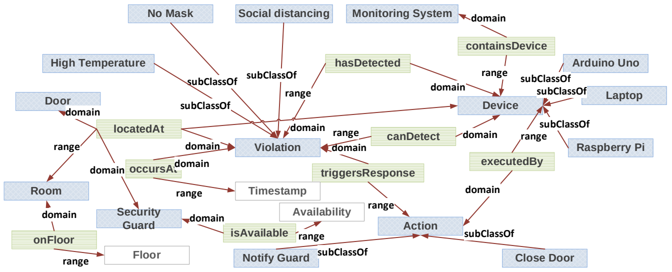


Fig. 4. An excerpt from COVID-19 Indoor Safety Monitoring Ontology.

In Listing 4., an example of SPARQL query for simple location matching in case of Violation of a type *Social Distancing* violation leveraging *Floor* is given.

```
PREFIX cismo: http://www.example.com/CISMO/
SELECT ?sg
WHERE {
  GRAPH <http://www.example.com/example1> {
    ?sg cismo:locatedAt ?r1.
    ?v  cismo:locatedAt ?r2.
    ?v  rdf:Type cismo:SocialDistancing.
    ?r1 cismo:onFloor ?f1.
    ?r2 cismo:onFloor ?f2.
    ?sq cismo:isAvailable 1.
        FILTER(?f1=?f2)
        }
}
```

Listing 4. SPARQL query for finding the security guard from the same floor where social distancing violation occurred.

V. Experiments and Evaluation Results

For evaluation, the following, the following devices were used: a laptop equipped with Intel i7 7700-HQ quad-core CPU running at 2.80 GHz with 16GB of DDR4 RAM and 1TB HDD acting as Edge server, Raspberry Pi 2B (RPi 2B), Raspberry Pi 3 (RPi 3) and Arduino Uno Rev. 3 with both IR sensor and thermal camera. In Table I, the results of performance evaluation for various scenarios, devices and settings are given. The first column denotes the name of the scenario (mask detection, distance check, contactless temperature measurement). The second column shows the hardware configuration used in the experiment. Moreover, the third column is the frame size expressed as number of horizontal multiplied by vertical pixels. Furthermore, the next column shows performance results achieved for given configuration expressed as number of processed frames (fps) or measurements per second (mps). Finally, the last column represents the accuracy achieved for the observed scenario. It is expressed as average percentage of successfully detected cases for mask detection and social distancing, while in case of temperature measurement it is average measurement error for corresponding sensor.

TABLE I
EVALUATION RESULTS

Scenario	Device	Frame size [W x H]	Frame rate [fps/mps]	Accuracy
Mask detection	RPi 2B	640x480	0.48	84-91%
		320x240	1.71	
	RPi 3	640x480	0.76	
		320x240	2.83	
	Laptop	640x480	11.94	
		320x240	38.46	
Distancing check	RPi 2B	640x480	0.72	65-73%
		320x240	2.65	
	RPi 3	640x480	1.12	
		320x240	4.29	
	Laptop	640x480	16.77	
		320x240	61.17	
Temperature sensing	IR (MLX90614)	1	8	0.5°C
	Thermal camera (AMG8833)	8x8	2	2.5°C

Considering the results shown in Table I, it can be concluded that RPi 3 has better performance than RPi 2B. It can be explained by the fact that RPi 3 utilizes newer quad-core ARM Cortex-A53 running at 1200 MHz then RPi 2B's 900 MHz ARM Cortex-A7. However, they both have only 1GB of RAM and their performance is still much behind the laptop. For lower resolutions, all devices show better performance, as expected.

Comparing the mask detection and social distancing check performance, we see that the second is faster as it only uses one classifier (full body), while mask detection uses three of them (face, nose and mouth). In all social distancing check experiments, the performance for two persons was evaluated, while it is expected to reduce as the number of people within the camera view increases. The performance of distancing check varies together with distance of objects from camera, as it changes with respect to initially calculated ratio between pixels and meters. The accuracy of both computer vision scenarios increases with resolution, but the cost is paid with performance decrease. Despite the acceptable accuracy of mask detection algorithm, it is not designed to detect transparent masks and face shields, which is a potential drawback.

On the other side, regarding the contactless temperature measurement, we can see that thermal camera is less accurate more demanding for computation, as it includes 64 measurements (8x8 matrix). However, its main advantage over IR sensor is the ability to measure the temperature of several persons at once, but requires additional data processing.

Finally, when it comes to time needed to find the appropriate security guard and generate MQTT message, in all the experiments, it does not exceed 1 second.

VI. Conclusion and Future Work

According to the achieved results, the proposed solution is usable for its purpose under certain performance limitations (such as number of processed frames or measurements per second). Moreover, it relies on both open hardware and free software, being definite and desirable advantage for such systems.

In future, it is planned to experiment with various deep learning and computer vision frameworks for object detection on Raspberry Pi in order to achieve higher framerate. Moreover, we would like to extend this solution with environment sensing mechanisms for adaptive building air conditioning and ventilation airborne protection in order to reduce the spread of coronavirus indoors [4, 8, 24], especially during summer. Furthermore, we will consider the implementation of mechanisms for transparent face shield detection. Finally, the ultimate goal is to integrate the system presented in this paper with our framework for efficient resource planning during pandemic crisis [25] in order to enable efficient security personnel scheduling and mask allocation, together with risk assessment based on statistics about respecting the safety guidelines and air quality.

Acknowledgment

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

References

[1] Coronavirus Update (Live) [online]. Available on: https://www.worldometers.info/coronavirus/, last accessed: 11/07/2020.

[2] P. Zhai et al., "The epidemiology, diagnosis and treatment of COVID-19", International Journal of Antimicrobial Agents vol. 55 issue 5, May 2020, 105955, pp. 1-13, 2020. https://doi.org/10.1016/j.ijantimicag.2020.105955

[3] P. Dawson et al., "Loss of Taste and Smell as Distinguishing Symptoms of COVID-19", Clinical Infectious Diseases June 2020, pp. 1-4, 2020. https://doi.org/10.1093/cid/ciaa799

[4] L. Morawska, "How can airborne transmission of COVID-19 indoors be minimised?", Environment International vol. 142, September 2020, 105832, pp. 1-7, 2020. https://doi.org/10.1016/j.envint.2020.105832

[5] T. Galbadage, B. Peterson, R. Gunasekera, "Does COVID-19 Spread Through Droplets Alone?", Frontiers in Public Health, vol. 8, April 2020, pp. 1-4, 2020. https://doi.org/10.3389/fpubh.2020.00163

[6] Coronavirus Incubation Period [online]. Available on: https://www.worldometers.info/coronavirus/coronavirus-incubationperiod/ , last accessed: 11/07/2020.

[7] D. Oran, E. Topol, Prevalence of Asymptomatic SARS-CoV-2 Infection: A Narrative Review, Annals of Internal Medicine, June 2020, pp. 1-7, 2020. https://doi.org/10.7326/M20-3012

[8] T. Dbouk, D. Drikakis, "On respiratory droplets and face masks", Physics of Fluids 32, 063303, pp. 1-11, 2020. https://doi.org/10.1063/5.0015044

[9] Y. Song et al., "COVID-19 Treatment: Close to a Cure? – A Rapid Review of Pharmacotherapies for the Novel Coronavirus" [preprint], pp. 1-25, 2020. https://doi.org/10.20944/preprints202003.0378.v1

[10] V. Balachandar et al., "COVID-19: emerging protective measures", European Review for Medical and Pharmacological Sciences vol. 24 (6), pp. 3422-3425, 2020. https://doi.org/10.26355/eurrev_202003_20713

[11] Open Computer Vision [Online]. Available on: https://opencv.org/ , last accessed: 07/07/2020.

[12] N. Petrović, "Upravljanje multimedijalnim sistemom pomoću algoritma za detekciju lica na Raspberry PI arhitekturi u realnom vremenu", IEEESTEC – 8th Student Projects Conference, Niš, Serbia, pp. 21-24, 2015.

[13] P. Viola, M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 511-518, 2001.

[14] J. Davies, R. Studer and P. Warren, *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, John Wiley & Sons, 2006.

[15] A. Venceslau et al., "IoT Semantic Interoperability: A Systematic Mapping Study", ICEIS 2019, pp. 535-544, 2019. https://doi.org/10.5220/0007732605350544

[16] N. Petrović, Đ. Kocić, "Data-driven Framework for Energy-Efficient Smart Cities", Serbian Journal of Electrical Engineering, Vol. 17, No. 1, Feb. 2020, pp. 41-63. https://doi.org/10.2298/SJEE2001041P

[17] Z. Wang et al., "Masked Face Recognition Dataset and Application" [preprint], pp. 1-3, 2020. https://arxiv.org/pdf/2003.09093.pdf

[18] T. Meenpal, A. Balakrishnan, A. Verma, "Facial Mask Detection using Semantic Segmentation", 2019 4th International Conference on Computing, Communications and Security (ICCCS), pp. 1-5, 2020. https://doi.org/10.1109/CCCS.2019.8888092

[19] M. Kavitha, S. M. M. Roomi, K. Priya, K. B. Devi, "State model based face mask detection", International Journal of Engineering & Technology, 7 (2.22), pp. 35-38, 2018.

[20] R. Biswas, A. Roy, "Real Time Temperature Graph using MATLAB and Arduino", International Journal of Engineering Research & Technology (IJERT) vol. 9 issue 5, pp. 624-625, 2020. https://doi.org/10.17577/IJERTV9IS050482

[21] M. J. Pramila, P. S. Shewta, "Wireless Temperature detector System using ARDUINO and IOT", International Journal of Computer Trends and Technology (IJCTT) vol. 67 issue 11, pp. 82-83, 2019. https://doi.org/10.14445/22312803/IJCTT-V67I11P113

[22] N. Petrovic, "Surveillance System Based on Semantic Video and Audio Annotation Leveraging the Computing Power within the Edge", XIV International SAUM 2018, pp. 281-284, 2018.

[23] How to calculate meters per pixel for a given camera? [online]. Available on: https://engineering.stackexchange.com/questions/32892/how-to-calculate-meters-per-pixel-for-a-given-camera, last accessed: 11/07/2020.

[24] R. V. Gomeseria, "Building Services Design to Prevent the Spread of COVID19", pp. 1-18, 2020. https://doi.org/10.17605/OSF.IO/BQM2F

[25] N. Petrovic, Dj. Kocic, "Framework for Efficient Resource Planning in Pandemic Crisis", CIIT 2020, pp. 1-6, 2020.

Data-Driven Mobile Applications Based on AppSheet as Support in COVID-19 Crisis

Nenad Petrović, Maša Radenković and Valentina Nejković

***Abstract*—In this paper, it is explored how data-driven mobile applications can contribute to battle against the COVID-19 pandemic. As proof of concept, two novel case studies are presented: indoor safety monitoring and resource planning during pandemic crisis. AppSheet platform for automated development based on Google Sheets as data source was used for implementation of corresponding multiplatform mobile applications.**

***Index Terms*—AppSheet; coronavirus; Google Sheets; mobile applications.**

I. Introduction

Since the beginning of this year, the world tackles the novel, flu-alike and highly infectious respiratory disease *COVID-19* caused by *SARS-Cov-2* virus (often referred to as *coronavirus*) [1]. It was discovered in Wuhan, China, but reached the other parts of the world in few weeks afterwards.

The most common symptoms of this disease are: fever, short breath, dry cough, fatigue, loss of taste and smell [2]. It is transmitted both directly through respiratory droplets and indirectly, via surfaces [3]. Incubation period varies from case to case, but could be quite long [4]. However, even asymptomatic persons can spread the disease [5], while death rate is quite high when it comes to older population and persons with chronic diseases [6], which makes the situation almost impossible to handle. The adoption of face masks and hand sanitizers has shown promising results in case of disease spread reduction [7]. However, the main issue is the lack of approved vaccine and medication [8].

Due to all the mentioned facts, many protection and safety measures were taken by governments in order to reduce the disease spread, such as obligatory indoor mask wearing, social distancing, quarantine, self-isolation, working from home, limiting citizens' movement and tourism, prohibition and cancellation of public events and gatherings [9]. Apart the huge number of lost lives, the pandemic has also caused catastrophic financial losses and stagnation of economy [10], leading towards overall global crisis.

When it comes to the global battle against COVID-19 and reduction of pandemic crisis consequences, it was concluded that state of the art technologies: artificial intelligence, blockchain, IoT and smartphone applications have huge potential. The main role of IoT devices is to collect sensor data about persons with goal to provide early diagnosis, enforce safety or guideline rules or help to patients and persons in isolation [11, 12]. On the other side, artificial intelligence algorithms enable extract meaningful knowledge and patterns from the collected sensor data to enable predictions of future trends or detect certain state/behavior by classifying observations [13]. Moreover, blockchain can be used to reduce the spread of misinformation during pandemics and trusted patient number tracking by ensuring authenticity or even transaction execution between the involved parties for purpose of resource exchange [14, 15, 16]. Finally, mobile applications offer interface to the services provided by previous enabling technologies for timely adoption among the targeted user groups [17, 18].

In this paper, it is explored how AppSheet[1] platform can be used for rapid mobile application development in order to support the society during COVID-19 pandemics. As outcome, two case studies making use of the previous enabler technologies are presented: 1) indoor safety monitoring app for security guards based on computer vision-enabled IoT solution 2) resource planning app for city representatives relying on AI-based predictions and blockchain for resource exchange.

II. Background and Related Work

A. AppSheet

AppSheet is an online development platform which enables easy creation and distribution of mobile, tablet and web applications starting from cloud data sources, such as spreadsheets and databases without any coding. Its first version was released as independent project in 2014, while it was acquired by Google and became part of Google Cloud[2] in January 2020. It mainly aims business use cases, such as project management, customer relationship management and personalized reporting. AppSheet analyzes the structure of provided data sources and automatically generates the views within the application. However, the user is also able to customize the generated user interface by showing or hiding particular table columns, grouping or sorting data, creating additional views and inserting computed values with respect to defined formulas. Apart from widely used textual and numeric types, it gives the ability to embed the images and videos within the user interface as well. AppSheet is free to use for up to 10 beta/test users, while monthly fee has to be paid for commercial usage and larger user base. When it

Nenad Petrović, Maša Radenković and Valentina Nejković are with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mails: nenad.petrovic@elfak.ni.ac.rs, masa.radenkovic@elfak.rs, valentina.nejkovic@elfak.ni.ac.rs).

[1] https://www.appsheet.com/

[2] https://cloud.google.com/appsheet

comes to more sophisticated features, the free version of AppSheet supports embedding value prediction and optical character recognition (OCR), while premium features include sentiment analysis, anomaly detection, data clustering and many other commonly used machine learning techniques. There are some limitations, such as minimal number of data rows needed for application creation (which is 4) and prediction training (25 observations). In order to run software created using AppSheet on mobile devices (Android or iOS), it is necessary to download and install the client application from corresponding online store which serves as container for user-created applications. However, active internet connection for their usage is required. In Fig. 1, an overview of typical application creation workflow (starting from Google Sheets[3] as data source) using AppSheet is given.

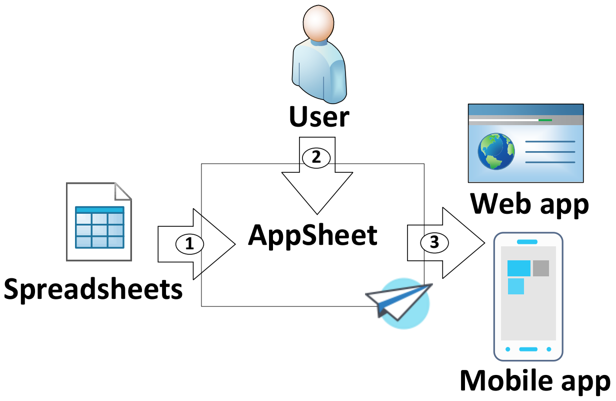


Fig. 1. Overview of typical AppSheet workflow: 1-Import data sources 2-User customizations 3-Generated output.

AppSheet-based mobile applications have been approved as effective solution across many industrial areas so far. In [19], AppSheet was used for real-time online monitoring of on-shelf availability in case of fast-moving consumer goods (FMCG). On the other side, in [20], it was used for water management application that calculates plant water consumption within given area. When it comes to healthcare, in [21], AppSheet was leveraged for development of electronic record keeping system at a pediatric clinic.

In this paper, AppSheet is adopted for COVID-19 support mobile application development, as it enables rapid prototyping and delivery of data-driven mobile applications, which is crucial in pandemic situation. As unexpected changes (such as dramatic rise of new disease cases) might occur quickly, it is of utmost importance to provide support services and respond to them timely in order to reduce the possible consequences [18]. Moreover, another advantage of AppSheet is multiplatform support without additional efforts and modifications. Finally, cross platform mobile development frameworks such as Xamarin require solid programming skills, while AppSheet enables easy application creation by domain experts.

B. *Mobile support applications in COVID-19 crisis*

In [22], a mobile application that allows people who are in self-isolation to ask for help and enable a fast, secure support by the community within 15 miles radius was presented. It was implemented using Flutter framework and supports both Android and iOS. On the other side, the AI-enabled mobile application presented in [23] aims to reduce the spread of COVID-19 disease by integrating the capabilities of user tracking, notification of the infected persons' employment/education institution and spread prediction based on the recorded movement locations. Furthermore, the work presented in [24] presents the computer vision approach which detects if face mask is worn properly using the image taken by smartphone camera. In [25], a mobile app for preliminary COVID-19 diagnosis based on cough samples was shown. In [26], an approach for on-device COVID-19 screening using snapshots of chest X-ray targeting smartphones was introduced. Finally, [27] discusses how mobile applications leveraging AI and AR can be used to measure athlete's performance without any physical contact with coach during the pandemic crisis. In Table I, overview of relevant COVID-19 support mobile applications with respect to their use cases is provided.

TABLE I
COVID-19 SUPPORT MOBILE APPLICATIONS

Application	**Use case**
C-19 C	Help in self-isolation
BreakTheChain	Spread prediction
CheckYourMask	Mask detection/check
AI4COVID-19	Cough-based diagnosis
COVID-MobileXpert	Chest X-ray screening

However, in this paper, the focus is on COVID-19 pandemic-related use cases not covered by the existing literature, mainly resource planning and indoor safety monitoring, relying on the work from [16] and [28].

III. Case Studies

A. *Indoor COVID-19 Safety Monitoring Security Guard App*

During the COVID-19 pandemic, it is very important for institutions and organizations to respect the COVID-19 safety rules and guidelines in order to reduce the disease spread, especially indoors [29].

This case study builds upon the work from [28], making use of IoT-based computer vision system that consists of Raspberry Pi devices equipped with camera. Each Raspberry Pi is located at a different location inside a building (main entrance or specific room entrance, for example) and is used to observe whether the passengers respect COVID-19 indoor safety guidelines: wearing masks and social distancing. For that purpose, computer vision algorithms are performed on-board in order to process the images coming from camera stream. In case that violation of safety rules is detected, then the Google Sheets API for Python[4] is used to append new data

[3] https://www.google.com/sheets/about/

[4] https://developers.google.com/sheets/api/quickstart/python

to the corresponding spreadsheet used as a log of rule violations at different locations within the building. Moreover, the camera image where violation was detected is uploaded to server and its URL is also included within the log together with the timestamp of current time. Apart from that, the corresponding security guard for a given location is determined using the remote call to the edge server and procedure described in [28]. This information is also included in the spreadsheet log, so the security guards can be notified which of the reported rule violations are under their responsibility. In order to keep track of the violations, security guards use mobile application developed using AppSheet, based on the data available within the violation log spreadsheet. In Fig. 2, overview of this case study is given.

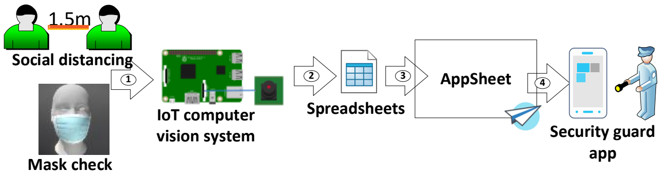


Fig. 2. COVID-19 indoor safety mobile application overview: 1-Camera stream analysis 2-Insert rule violation record 3-Update app 4-Notify security guard.

In Listing 1, the role of IoT system in context of security guard mobile application for violation log creation is described as pseudocode.

```
Input: image, sheet_id, location
Output: violations
Steps:
distance_violation:=SocialDistanceCheck(image);
mask_violation:=MaskCheck(image);
if(distance_violation==false and mask_violation==false)
    violations:="No violation";
else
    if(mask_violation==true)
        violations:=violations+"No mask or worn improperly";
    if(distance_violation==true)
        violations:=violations+"Social distancing not satisfied";
    image_url:=UploadImage(image);
    guard_id:=RemoteCall.GetFreeGuard(location);
    t:=Now();
    WriteToSheet(sheet_id, image_url, violations, location, t, guard_id);
end if;
return violations;
end.
```

Listing 1. IoT indoor safety monitoring system modifying log using Google Sheets API in pseudocode.

In Fig. 3, the use case diagram of this application is shown.

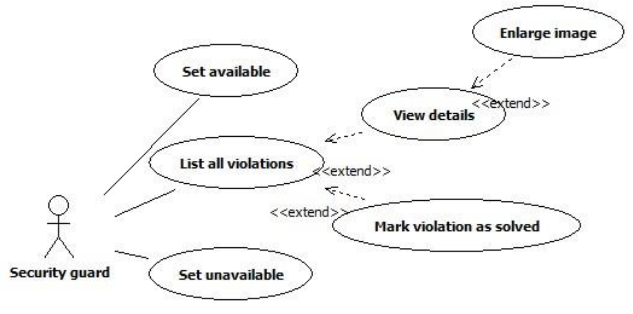
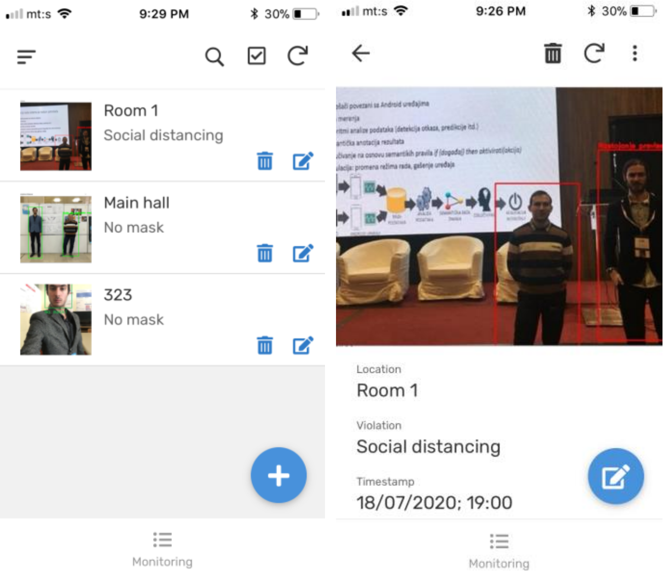


Fig. 3. Indoor safety security guard App use cases.

In Fig. 4, screenshots of security guard AppSheet mobile app run on iPhone SE is given. The first view (a) shows the list of detected violations within the building, while the second one (b) shows details for the selected rule violation. For each violation, it is possible to see the enlarged image, time/date when it occurred together with the location where it was detected and which device performed its detection. Once security guard warns the persons breaking the safety rules, the violation event can be checked as resolved and removed from pending list. Moreover, security guards can set their availability status to determine whether they will be taken as candidates by the server's algorithm [28] for patrol task allocation or no.



Room 1
Social distancing

Main hall
No mask

323
No mask

Location
Room 1

Violation
Social distancing

Timestamp
18/07/2020; 19:00

Fig. 4. Indoor safety security guard AppSheet iOS mobile app screenshots: (a) list of detected violations (b) detailed violation view.

In Table II, the structure of the underlying Google Sheet used for AppSheet application generation is illustrated.

TABLE II
INDOOR SAFETY APPLICATION GOOGLE SHEET STRUCTURE

Location	*Device*	*Violation*	*T*	*Guard id*	*Image url*	*Solved*

B. Resource Planning Mobile Application

In pandemic, it is of utmost importance to plan resources efficiently, but timely – either human, material resources or equipment. The protection of health and the economy of a country are linked tightly [30].

This mobile app follows up the framework for efficient city resource planning during pandemic crisis from [16]. Its target users are city administration/financial representatives. For each city, relevant resources in context of pandemic crisis are considered [16]: budget, available hospital places, test kits, face masks, ambulance vehicles and doctors. Moreover, it is possible to see the detailed information about availability of resources and their prices.

For each day, the number of predicted new COVID-19 cases and deaths are taken into account in order to determine whether the available city resources are sufficient to satisfy the current demands. For daily case predictions, several factors are considered, such as historical data about cases in the considered city, weather parameters, safety measures applied within the city (lockdown or limited movement, gathering prohibitions, restaurant/bar working hours). The first two parameters are automatically updated daily, leveraging the publicly available online data from services, such as [31], [32] and [33], while the others are entered manually by users. Moreover, the new case predictions are further used within the formulas for resource prediction as described in [16]. The prediction mechanism completely relies on AppSheet whose minimum number of observations for training is 25. Moreover, calendar view of daily predictions is also generated using AppSheet. Finally, city representatives can make offers to other cities for resource exchange. If offer is accepted, resource exchange between the cities relying on blockchain and smart contracts will be performed building upon the work presented in [16] and [34]. In Fig. 5, an overview of described applications is given.

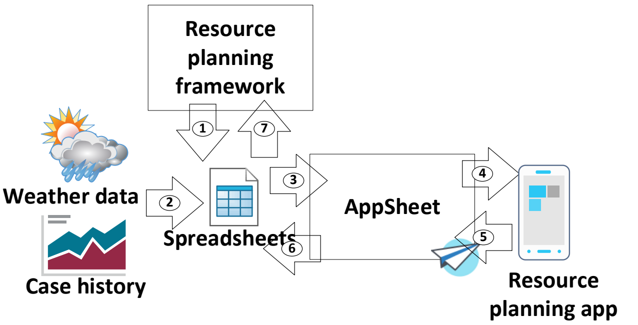


Fig. 5. COVID-19 resource planning mobile app: 1-City resource model 2-Online weather data and case history 3-Import spreadsheet 4-Generated data view 5-User input parameter 6-Predictions 7-Import spreadsheet into resource planning framework.

In Fig. 6, an overview of resource planning app use cases is given.

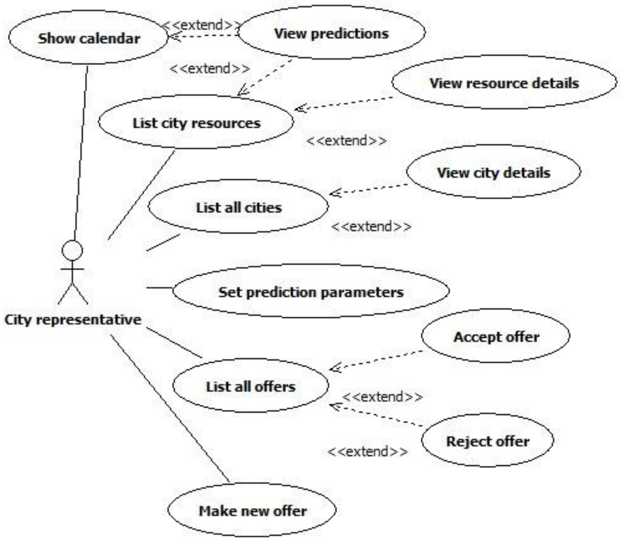
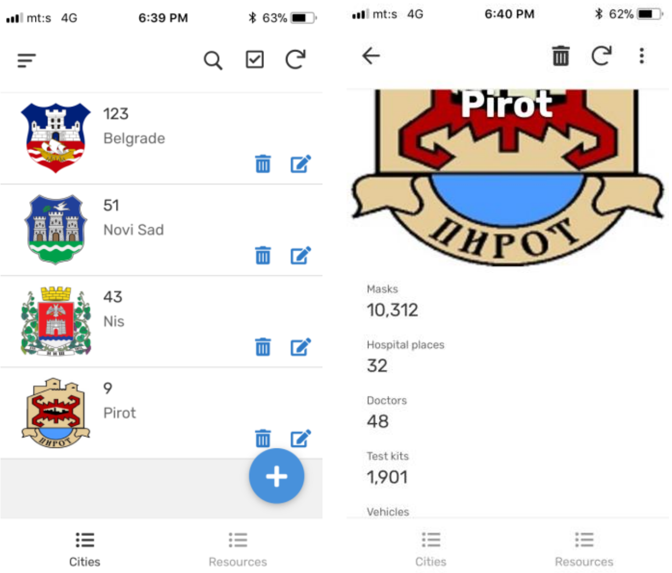


Fig. 6. City representative resource planning mobile app use case diagram.



mt:s 4G 6:39 PM 63%

123
Belgrade

51
Novi Sad

43
Nis

9
Pirot

Cities
Resources

mt:s 4G 6:40 PM 62%

Pirot
ПИРОТ

Masks
10,312

Hospital places
32

Doctors
48

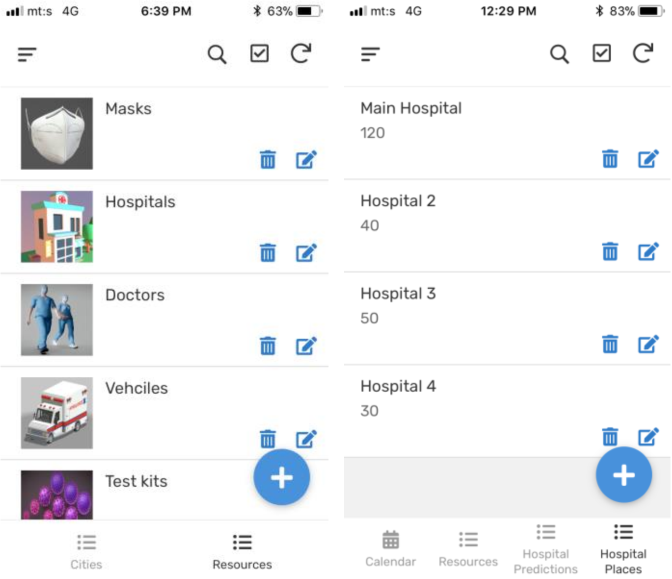
Test kits
1,901

Vehicles

Cities
Resources

Fig. 7. Resource planning AppSheet iOS mobile app screenshots: (a) city overview (b) city resource view.

Fig. 7. shows a screenshot of city resource views taken on iPhone SE. The first screen (a) shows the list of available cities together with number of new cases that day. The second screen (b) shows summary of available resources for each of the cities. Furthermore, Fig. 8 displays the details the list of available resource types and details for each of them (b). In this case, the number of available places in hospitals for the selected city is shown.



mt:s 4G 6:39 PM 63%

Masks

Hospitals

Doctors

Vehciles

Test kits

Cities
Resources

mt:s 4G 12:29 PM 83%

Main Hospital
120

Hospital 2
40

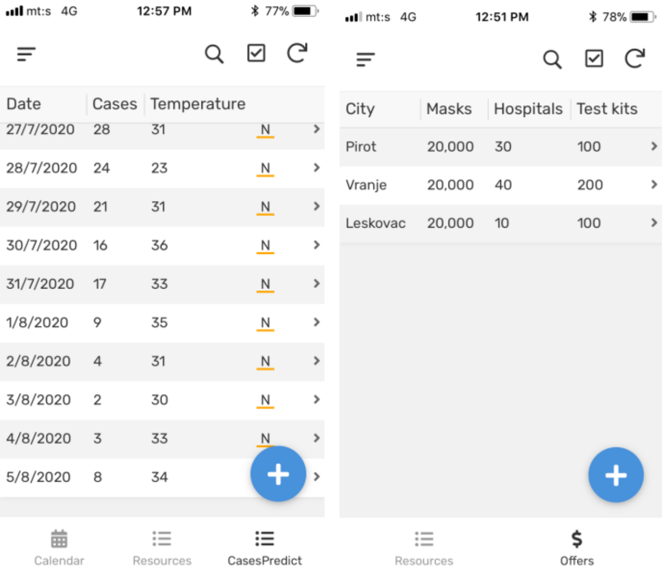
Hospital 3
50

Hospital 4
30

Calendar
Resources
Hospital Predictions
Hospital Places

Fig. 8. Resource planning AppSheet iOS mobile app resource view screenshots: (a) resource list (b) resource details.

In Fig. 9, the screenshots of case prediction view (a) and offer list (b) are shown. When it comes to prediction view (a), users can provide the input parameters for case predictions (apart from those that are obtained from online sources automatically) that are further used to determine the daily demand of resources. On the other side, in offers view (b), user can see all the resource trading offers for the city under responsibility and accept/reject them.



Date	Cases	Temperature	
27/7/2020	28	31	N
28/7/2020	24	23	N
29/7/2020	21	31	N
30/7/2020	16	36	N
31/7/2020	17	33	N
1/8/2020	9	35	N
2/8/2020	4	31	N
3/8/2020	2	30	N
4/8/2020	3	33	N
5/8/2020	8	34	

City	Masks	Hospitals	Test kits
Pirot	20,000	30	100
Vranje	20,000	40	200
Leskovac	20,000	10	100

Fig. 9. (a) Case prediction screenshot (b) Offer list screenshot.

A brief summary of the main Google Sheets used for generation of resource planning AppSheet mobile application is given in Table III.

TABLE III
RESOURCE PLANNING APPLICATION GOOGLE SHEETS STRUCTURE OVERVIEW

Sheet	***Col1***	***Col2***	***Col3***	***Col4***
Resources	City [text]	Budget [num]	Hospital places [num]	Test kits [num]
Hospitals	City [text]	Name [text]	Free places [num]	Cost [num]
Test kits	City [text]	Number [num]	Type [text]	Cost [num]
Prediction	Date [dd/mm/yy]	Cases [num]	Temperature [°C]	Lockdown [y/n]
Offer	City [text]	Hospitals [num]	Test kits [num]	Price [num]

IV. EXPERIMENTS AND EVALUATION RESULTS

For purpose of evaluation, the following, the following devices were used: a laptop equipped with Intel i7 7700-HQ quad-core CPU running at 2.80GHz with 16GB of DDR4 RAM and 1TB HDD acting as Edge server, iPhone SE 2016 (dual-core, 2GB of RAM) as representative of iOS smartphones and Xiaomi Redmi Note 7 (octa-core, 3GB of RAM) as representative of Android smartphones. As active internet connection was required to run the apps, 4G network with 50Mbps download/upload speed was used.

Various aspects of evaluation were considered: application development time speed-up (compared to equivalent Android application developed in Android Studio [28]), application delivery time (how much it takes to see the application modifications without re-installing), storage required for AppSheet client, RAM memory utilization (for Android) and average prediction accuracy of AppSheet's predictor in case of new cases and resource demand predictions. In Table IV, the results achieved during the evaluation are given.

TABLE IV
EVALUATION RESULTS

Aspect	**Value**	
Development speed-up [times]	12 times	
Delivery time [s]	iPhone	Redmi
	4.12	3.26
Client app size [MB]	iOS	Android
	.ipa:59 total:96	.apk:13 total:77
RAM utilization [MB]	Case A. 132	Case B. 184
Prediction accuracy [%]	85.6%	

According to the achieved results, the proposed approach of mobile application generation using AppSheet taking Google Sheet as an input significantly speeds up the mobile application development compared to native solutions. For evaluation, the case study (A) was considered as native Android mobile app was already developed in [28]. Using AppSheet, around 30 minutes were needed, while it took around 6h to develop similar application from scratch in Android Studio. However, the native application uses Message Queuing Telemetry Transport (MQTT) protocol and libraries for message exchange, while AppSheet solution logs the violations within spreadsheet.

When it comes to delivery time, it was faster in case of Android device, as it is equipped with more powerful CPU and larger RAM, while AppSheet client is larger in case of iOS. Furthermore, resource planning tool occupies more RAM than the indoor safety monitoring application, as it has much more views and features, such as value prediction. However, indoor safety monitoring application is much heavier than native application in terms of storage required and RAM occupation, as native Android indoor safety monitoring app was just 6MB, while only around 50MB of RAM was required.

Finally, when it comes to accuracy of case predictions, it was higher than in case of [16] which was around 72%, as much more data was available (whole May, June and July) and AppSheet's prediction module is highly optimized, but still fast, as training took just several seconds.

V. CONCLUSION AND FUTURE WORK

According to the achieved results for case studies, the proposed approach of AppSheet-based COVID-19 support mobile application development has huge potential, as it provides rapid creation, fast delivery, relatively low hardware requirement requirements and solid set of features at the same time. Unlike state-of-the-art cross platform frameworks, it requires almost no coding, while it provides more advanced user interfaces, data visualization and AI-enabled capabilities compared to mobile version of Google Sheets. In one of the case studies, the prediction module was used to estimate the daily number of new cases and according to that, the city resource demand with satisfiable accuracy.

However, in future, it is planned to perform tighter integration with blockchain infrastructure in order to enable secure logging of the events that occur, case data and resource availability leveraging smart contracts. Finally, the integration with framework for smart contract verification based on semantic technology [35] is also considered. Moreover, apart from COVID-19 use cases, we plan to adopt AppSheet applications for custom fitness training generation based on user parameters (height, weight and other details) in future.

Acknowledgment

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

References

[1] V. C. Bala, P. Kumar, "Review on COVID-19: Rise of SARS-CoV-2 Pandemic Outbreak", Borneo Journal of Pharmacy, 3(Special-1), pp. 103-120, 2020. https://doi.org/10.33084/bjop.v3iSpecial-1.1412

[2] P. Zhai et al., "The epidemiology, diagnosis and treatment of COVID-19", International Journal of Antimicrobial Agents vol. 55 issue 5, May 2020, 105955, pp. 1-13, 2020. https://doi.org/10.1016/j.ijantimicag.2020.105955

[3] P. Dawson et al., "Loss of Taste and Smell as Distinguishing Symptoms of COVID-19", Clinical Infectious Diseases June 2020, pp. 1-4, 2020. https://doi.org/10.1093/cid/ciaa799

[4] Coronavirus Incubation Period [online]. Available on: https://www.worldometers.info/coronavirus/coronavirus-incubationperiod/ , last accessed: 11/07/2020.

[5] D. Oran, E. Topol, Prevalence of Asymptomatic SARS-CoV-2 Infection: A Narrative Review, Annals of Internal Medicine, June 2020, pp. 1-7, 2020. https://doi.org/10.7326/M20-3012

[6] Y. V. Chudasama, "Impact of COVID-19 on routine care for chronic diseases: A global survey of views from healthcare professionals", Diabetes & Metabolic Syndrome: Clinical Research & Reviews, Volume 14, Issue 5, pp. 965-967, 2020. https://doi.org/10.1016/j.dsx.2020.06.042

[7] T. Dbouk, D. Drikakis, "On respiratory droplets and face masks", Physics of Fluids 32, 063303, pp. 1-11, 2020. https://doi.org/10.1063/5.0015044

[8] Y. Song et al., "COVID-19 Treatment: Close to a Cure? – A Rapid Review of Pharmacotherapies for the Novel Coronavirus" [preprint], pp. 1-25, 2020. https://doi.org/10.20944/preprints202003.0378.v1

[9] V. Balachandar et al., "COVID-19: emerging protective measures", European Review for Medical and Pharmacological Sciences vol. 24 (6), pp. 3422-3425, 2020. https://doi.org/10.26355/eurrev_202003_20713

[10] M. Nicola et al., "The socio-economic implications of the coronavirus pandemic (COVID-19): A review", International Journal of Surgery, Volume 78, June 2020, pp. 185-193, 2020. https://doi.org/10.1016/j.ijsu.2020.04.018

[11] R. Vaishya, M. Javaid, I. H. Khan, A. Haleem, "Artificial Intelligence (AI) applications for COVID-19 pandemic", Diabetes & Metabolic Syndrome: Clinical Research & Reviews, Volume 14, Issue 4, 2020, pp. 337-339, 2020. https://doi.org/10.1016/j.dsx.2020.04.012

[12] R. P. Singh, M. Javaid, A. Haleem, R. Suman, "Internet of things (IoT) applications to fight against COVID-19 pandemic", Diabetes & Metabolic Syndrome: Clinical Research & Reviews, Volume 14, Issue 4, 2020, pp. 521-524, 2020. https://doi.org/10.1016/j.dsx.2020.04.041

[13] M. Nasajpur et al., "Internet of Things for Current COVID-19 and Future Pandemics: An Exploratory Study" (preprint), pp. 1-19, 2020. https://arxiv.org/abs/2007.11147

[14] D. Marbouh et al., "Blockchain for COVID-19: Review, Opportunites and a Trusted Tracking System" (preprint), pp. 1-17, 2020. https://doi.org/10.36227/techrxiv.12609344.v2

[15] T. Alam, "Internet of Things and Blockchain-Based Framework for Coronavirus (COVID-19) Disease" (preprint), pp. 1-9, 2020. https://doi.org/10.20944/preprints202007.0642.v1

[16] N. Petrovic, Dj. Kocic, "Framework for Efficient Resource Planning in Pandemic Crisis", CIIT 2020, pp. 1-6, 2020.

[17] L. C. Ming et al., "Content Analysis and Review of Mobile Health Applications on COVID-19", JMIR mHealth and uHealth. 30/06/2020:19796 (forthcoming/in press). https://doi.org/10.2196/19796

[18] D. Drew et al., "Rapid implementation of mobile technology for real-time epidemiology of COVID-19", Science Vol. 368, Issue 6497, pp. 1362–1367, 2020. https://doi.org/10.1126/science.abc0473

[19] N. Harraz, "Online Monitoring of On-Shelf Availability", International Conference on Industrial Engineering and Operations Management 2018, pp. 450-461, 2018.

[20] P. Samutrak, C. Kangvaravoot, "Application to Calculate Potential Evapotranspiration", International Journal of Applied Computer Technology and Information Systems: Volume 7, No.1, pp. 35-40, 2017.

[21] D. S. Wijesekara et al., "Developing an electronic record keeping system at a paediatric clinic in Colombo South Teaching Hospital, Sri Lanka", Sri Lanka Journal of Child Health, 49(2), pp.116–124, 2020.

[22] E. Ogunshile, K. Phung, S. Odongo, "C-19 C -A mobile application to support the community during Covid-19 lockdown", CONISOFT 2020, pp. 1-9, 2020.

[23] J. Vanathi, G. SriPradha, "BreakTheChain: A Proposed AI powered Mobile Application Framework to handle COVID-19 Pandemic", Alochana Chakra Journal, Volume IX, Issue IV, April 2020, pp. 108-114, 2020.

[24] K. Hammoudi, A. Cabani, H. Benhabiles, M. Melkemi, "Validating the correct wearing of protection mask by taking a selfie: design of a mobile application "CheckYourMask" to limit the spread of COVID-19" (preprint), pp. 1-5, 2020. https://hal.archives-ouvertes.fr/hal-02614790

[25] A. Imran et al., "AI4COVID-19: AI enabled preliminary diagnosis for COVID-19 from cough samples via an app" (pre-proof), Informatics in Medicine Unlocked, pp. 1-30, 2020. https://doi.org/10.1016/j.imu.2020.100378

[26] X. Li, C. Li, D. Zhu, "COVID-MobileXpert: On-Device COVID-19 Screening using Snapshots of Chest X-Ray" (preprint), pp. 1-14, 2020. https://arxiv.org/abs/2004.03042

[27] J. Lim, "Measuring sports performance with mobile applications during the COVID-19 pandemic", SPSR 2020, 103, v1, pp. 1-3, 2020.

[28] N. Petrović, Đ. Kocić, "IoT-based System for COVID-19 Indoor Safety Monitoring" (preprint), IcETRAN 2020, pp. 1-6, 2020.

[29] L. Morawska, "How can airborne transmission of COVID-19 indoors be minimised?", Environment International vol. 142, September 2020, 105832, pp. 1-7, 2020. https://doi.org/10.1016/j.envint.2020.105832

[30] P. Vanini, "Protection of the Population and the Economy in a Pandemic" (preprint), pp. 1-21, 2020.

[31] Statistika COVID-19 u Srbiji [online]. Available on: https://covid19.data.gov.rs/, last accessed: 29/07/2020.

[32] Zaraženo u danu [online]. Available on: https://covid19.data.gov.rs/infected, last accessed: 29/07/2020.

[33] Serbia weather conditions [online]. Available on: https://www.accuweather.com/en/rs/serbia-weather, last accessed: 29/07/2020.

[34] N. Petrović, Đ. Kocić, "Data-driven Framework for Energy-Efficient Smart Cities", Serbian Journal of Electrical Engineering, Vol. 17, No. 1, Feb. 2020, pp. 41-63. https://doi.org/10.2298/SJEE2001041P

[35] N. Petrović, M. Tošić, "Semantic Approach to Smart Contract Verification", Facta Universitatis Series Automatic Control and Robotics 19(1), pp. 21-37, 2020.

Meteorological Data Aggregation System for Application in Agriculture 2.0

Balša Lazarević and Tomo Popović, *Senior Member*, *IEEE*

***Abstract*—Digital transformation has given numerous new possibilities for improvement of all industries through creation of new business processes or changes to existing ones. Agriculture, specifically, has experienced a "rebirth" through the use of digital technologies such as sensor networks, drones, satellites and similar, which has received a name Agriculture 2.0. Meteorological data management systems found their use in agriculture decision supporting solutions, as well. However, there are some challenges, these solutions are only as good as the data made available to them and implementing a system for acquiring quality data may be very expensive. The research described in this paper aims to solve these issues by using multiple publicly available free weather data sources and analyzing it to get the best possible results. The proposed system builds on acquired and analyzed data to provide users with features such as reliable notifications and machine/equipment automation.**

***Index Terms*—agriculture 2.0; data analytics; digital transformation; meteorological data; precision agriculture; weather data aggregation.**

I. Introduction

Digital transformation is a comprehensive process of using digital technologies to create new or change existing business processes, culture and user experiences and generally change the way of doing business through the use of digital technologies and the improvements they enable [1,2].

The digital transformation began when full advantages of digitalization were recognized and started being used. It can be best seen in the changes of work of customer services - the processes of accepting applications, searching for relevant data/responses and resolving the issues were completely transformed when it was "discovered" that data no longer had to be searched in opaque dusty folders and cabinets, but could be found in a couple of mouse and keyboard clicks. That, basically, represents the digital transformation, it is not a simple improvement in efficiency of existing methods by using digitalization, but it is understanding and using the advantages and potentials of information technology for creating new business methods and processes [3].

By the beginning of the 21st century, most farmers ran their business processes manually, relying primarily on experience and tradition. At the beginning of the century, that began to change. The application of digital technologies has automated a large number of processes, and it has enabled much better monitoring and better decision making for many more [4]. Innovations such as a milking robot, automation of machines for digging the soil, planting, pruning plants, or harvesting enable much easier more productive work for farmers [5].

In the last couple of decades, base (navigation) stations for precise agricultural condition have appeared, which enabled independent navigation for machines in the fields, and along with them, meteorological stations that directly provided meteorological data to the machines started being used. That's when the real digital transformation in agriculture began. With increased precision and the ability to process huge amounts of data, machines were able to take over more sensitive jobs and operate independently of humans [6].

The majority of people know that weather has an important impact on the agricultural industry. Crops need moisture, heat and sun to succeed. What is less obvious is how some weather information can influence farmers' business decisions, helping them plan efficiently, minimize costs and, as a result, increase yields and earnings. Farmers have to make weather-related decisions on a daily basis and there are four basic areas of agriculture where weather conditions are mainly used:

- Crop growth / irrigation,
- Frost prevention,
- Pest and disease control,
- Crop cultivability,
- Fertilization.

During the year, farmers make small but frequent decisions about their crops and the cumulative effect of the financial implications of those decisions can be significant, especially when it comes to producers who have thousands of hectares planted. For them, unforeseen weather conditions represent a much higher risk and cost than the cost of using accurate and reliable weather data [7]. In recent years, digital transformation of agriculture and solutions for precision agriculture are expected to play a key role in developing countries [8]. Additional motivation for this research is that precision agriculture is being explored and evaluated for the use in Montenegro [9,10], and Agriculture and information technology are recognized as strategic priorities in the Smart Specialisation Strategy of Montenegro 2019-2024 [11].

II. Background and Approach

The focus of this research is creation of a meteorological data management system capable of providing accurate meteorological data, both current data and forecasts, in a way that enables farmers to make informed decisions, whether they do it themselves, manually, or whether these decisions

Balša Lazarević is with Faculty for Information Systems and Technologies, University of Donja Gorica, Oktoih 1, 81000 Podgorica, Montenegro (e-mail: balsa.lazarevic@udg.edu.me).

Tomo Popović is with Faculty for Information Systems and Technologies, University of Donja Gorica, Oktoih 1, 81000 Podgorica, Montenegro (e-mail: tomo.popovic@udg.edu.me).

are made automatically by machines. The paper describes the key elements of the information system that provides quality and accurate meteorological data, with the possibility of sending notifications to users and direct connection with agricultural machines in order to automate them.

A. Needs for Precise Meteorological Data

Accurate meteorological data are necessary for any studies related to climate trends and meteorological events. Automatic meteorological stations are one of the best instruments in meteorology today and enable fast and efficient measurements, as well as measurements in remote areas [12]. In addition to using the data itself, the goal of collecting raw, current meteorological data is also to analyze them and make predictions based on them for the near or distant future. Currently, there are a handful of institutions in the world, but also companies that deal with climate forecasts, each using different methods of analysis and forecasting models. Certainly, all of these models also have different levels of precision and companies are competing with each other to create the most accurate model. They are doing this not only out of prestige or scientific curiosity (although yes to a certain extent), but as mentioned earlier, because predictions have a great influence on economic decisions in various industries, and primarily in agriculture [7,8].

B. Weather Data Collection

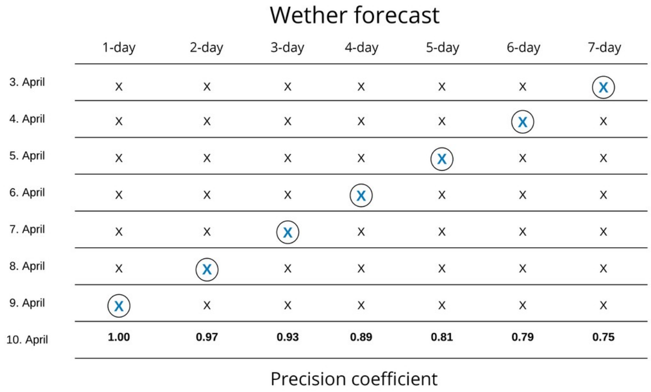
The process of analyzing meteorological data begins with the regular collection of current meteorological data for locations in the system. Data is retrieved at regular intervals, every hour, half hour, or more frequently, from several available sources. These sources are defined as publicly available APIs offered by weather data providers that provide weather data for locations based on GPS coordinates. In addition, weather data may be obtained from user owned devices equipped with sensors, either by polling the devices for data or devices sending the data to the system. The data collected from multiple sources is then standardized and formatted according to the structure of the table in the database, and finally entered into the database for aggregation and further processing. Even though sources provide the same data such as temperature, humidity, etc, they differ from each other in several technically important ways. As a rule, the "keys" of the data are always different, the daily temperature can be called *Temperature* by one source and *DailyTemperature* by another. In addition, there are often differences in the units in which values are displayed, precipitation can be displayed in centimeters but also inches, and differences can also be found in the data structure itself. Therefore, a different way of processing (parsing) and formatting data and their standardization is made for each source.

III. Implementation and Discussion

A. Weather Data Aggregation and Analytics

In parallel with the regular retrieval of current meteorological data, for each location the system is retrieving meteorological forecasts from multiple sources for the next 7 days on a daily basis. It is at this step that it is necessary to perform data analysis. The first part of the process, forecasts collection and formatting, works like the process of collecting current measurement data. The second part of the process, analysis, is the one that differs. After collecting and formatting the data, instead of entering it directly into the database, precision coefficients must be added to the forecasts so that users know how reliable each forecast is.

In order to estimate the coefficient of precision of a forecast, we take the current weather data and compare them with the values that were predicted for this moment in the past (last 7 days). Here, precision analysis is implemented by taking one parameter, the maximum daily temperature, and for even more precise analysis, more or all data can be taken. Figure 1 shows the values taken from the previous 7 forecasts from the database. The precision coefficients and values on the graph are shown only illustratively.



Wether forecast

	1-day	2-day	3-day	4-day	5-day	6-day	7-day
3. April	x	x	x	x	x	x	ⓧ
4. April	x	x	x	x	x	ⓧ	x
5. April	x	x	x	x	ⓧ	x	x
6. April	x	x	x	ⓧ	x	x	x
7. April	x	x	ⓧ	x	x	x	x
8. April	x	ⓧ	x	x	x	x	x
9. April	ⓧ	x	x	x	x	x	x
10. April	**1.00**	**0.97**	**0.93**	**0.89**	**0.81**	**0.79**	**0.75**

Precision coefficient

Fig. 1. Analysis of the precision coefficient of a seven-day weather forecast

For each day of the forecast **n** that we want to estimate, we need to compare the present value on today's day **t** with the value that was forecast at time **t - n** for day **t**. So, to get the precision of the 1-day forecast **n = 1**, we need to take the **nth** value from the forecast series that was recorded on this day - n or **t - 1** and it is **[t - n]**. If we want to estimate the 2-day precision of the weather forecast **n = 2** on April 10, t = 10, we need to take the value **[t - n]**, **[10 - 2] = 8**, ie. in order to evaluate the precision of the forecast for April 12 on April 10 (two day forecast), we need to take the second value from the series of forecasts from April 8, which will be the predicted value for April 10, and compare it with the current measured value. For example, if on April 10 we want to evaluate the two-day precision, for April 12, we will take the current temperature value of 21 degrees Celsius, then, the second value from a series of forecasts from 2 days ago, April 8, which is 20 degrees, compare 20 / 21 and obtain a two-day forecast precision factor of 0.95 or 95%.

As stated, the implemented analysis takes only one parameter, the daily temperature, and for better results another parameter can be taken or even more of them depending on the needs.

For some activities in agriculture, in addition to temperature, precision in the occurrence and amount of precipitation, primarily rain, is more important. Therefore, along with the existing analysis, countless more analyzes can be created that will reflect the needs of certain users. Accuracy analysis can be created based on air pressure, rainfall precipitations, snow, or any combination thereof, and users would have the option to choose which analysis they want to monitor, or they could choose to track weighted accuracy coefficients based on all analyzes.

In addition to the analysis that can be performed on new data, sufficient data will be collected over time for meaningful analyzes of historical data. Also, adding more sources will increase the amount of available data exponentially and it will be possible to make detailed analyzes and comparisons among the sources themselves, that are much more advanced than precision analysis. Furthermore, this approach could be enhanced by applying artificial intelligence and machine learning to data sets of this size, it is certain that parallels and correlations between certain locations will be drawn as well as many other useful information as well.

B. Micro Level Weather Forecasts

In order to provide the user with the most accurate and high-quality meteorological data, it is necessary to implement the functionality of micro-locations, ie. that users are not limited to weather data for the city or municipality in which they are located, but that they can select the specific location they want on the map. This will allow users to choose, for example, their field or orchard and thus obtain highly localized data, rather than an average for a wider area.

At the beginning of the development of this functionality, a problem was encountered, and it is related to the unlimited number of possible user locations. The process begins with the user adding a location by entering its name and selecting a location on the map. When entering a location in the database, the location from the map is converted to latitude and longitude, which gives the following values - e.g. 42.435363, 19.261012. Each location has a certain latitude and longitude that consists of degrees, hours, minutes and seconds which equals to 2 numbers of 6 decimal places. The number of possible combinations of these two numbers is 32 septillion. In a hypothetical case where there is a database (or more of them) that could support so many locations, the question is whether it makes sense to make so many API requests to sources? The answer is no, and for three important reasons. The first reason is technical in nature, making so many requests would take too much time. The second reason is economical, all sources of meteorological data are services that provide meteorological data through the API for a certain amount of money. The cost of so many calls would simply not be worth the benefits. The third and main reason concerns the very nature of meteorological data. Although previous reasons could ultimately be technically and economically surpassed, the question arises whether they should be at all? Theoretically, it is possible to create millions of locations on one square kilometer, but that area is too small for the data to differ between locations at all. Therefore, it was decided to do a research that will show for which area perimeter the data does not change.

A short research was made with the task of determining the minimum distance between two points on the territory of Montenegro for which the retrieved data will differ.

The research was performed as follows:

- Several random locations on the territory of Montenegro were selected,
- The current measurement data for each of these locations were then retrieved, which is the basis on which the following data was compared.

The following was done for each location:

- Current measurement data 5 kilometers from the baseline was retrieved for each location;
- After that, the data 10 kilometers away from the baseline was retrieved for each location;
- And so it was continued, raising the distance by 5 km up to a distance of 30 kilometers. The following graph shows an example of data retrieval for several locations.

Figure 2 shows three locations marked with blue dots, and locations for which measured values were taken at a distance of 5 kilometers from each other as red dots. It was noticed that the data can differ significantly at the fifth point which is 25 kilometers away from the base location, but it was also noticed that in some cases the data can also differ at the fourth point at the distance of 20 kilometers. Therefore, it was decided to create a new type of data, Aggregate location, to which all User locations within a radius of 20 kilometers will be linked and for which meteorological data of one, aggregate location, will be served.

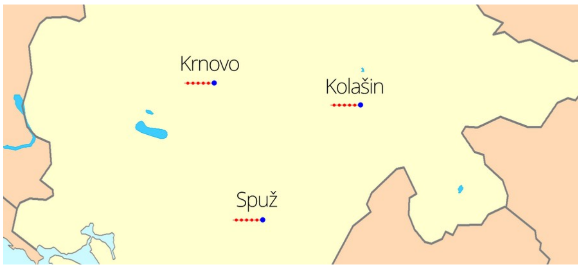


Fig. 2. Investigation of diversity of meteorological data by location

It was first decided to take a circle within a radius of 20 kilometers from the aggregate location as the surface of the aggregate location, but it was later switched into a square in order to be based on latitude and longitude. The final solution is that all user locations that fall under 0.1 degrees distance of latitude and longitude in relation to the aggregate location are to be linked to it. The distance of 0.1 degrees is taken because, calculated in kilometers, it amounts to 11 kilometers and 0.2 degrees diameter approximately fits a chosen distance of 20km.

As shown in Fig. 3, the process of adding user locations is marked with numbers. Under number 1 the first user location to the system is added. Since there is no location to which it can be linked, the first aggregate location with these coordinates is automatically added. The next is user location number 2 which is too far from aggregate location 1 and an aggregate location is automatically created for it and gets linked to it. After that, user locations 3 and 4 are entered. Since they are less than 0.1 degrees away from the aggregate location 1, they fall within its diameter and are linked to it in the database. These user locations will in future retrieve the meteorological data of the aggregate location 1. User location 5 cannot be linked to any aggregate location because it is too far away from them and an aggregate location is also created for it. Coincidentally, its area overlaps to some extent with aggregate location 1. This does not mean that new user locations that fall under the area that intersects the two locations will be linked to both locations, but they will belong to one of these two locations depending on distance.

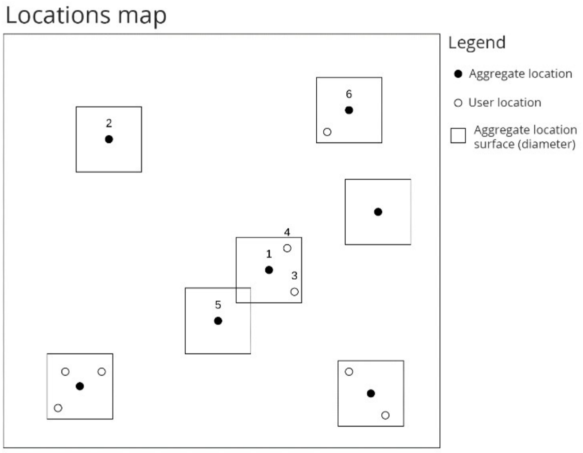


Fig. 3. Creating user defined locations for weather data aggregation

The implementation of this structure of locations enables the provision of quality and accurate meteorological data to end users for any of their micro locations, with a reasonable limitation in the form of aggregate locations.

C. Notification/Alarms

No matter how hard they try to give an accurate and precise weather forecast, meteorologists must always make certain generalizations, either on a spatial or temporal dimension. In addition to spatial differences between data, as happens within the same municipality, data also differs at the same location depending on the time of day. While we can get information on TV about the minimum and maximum daily temperature, or about the morning and evening weather conditions, the conditions may differ even within a time period such as "evening". Although more detailed meteorological data can be found on the Internet, mostly at the level of one hour, even this way of informing has two drawbacks - the first is that the weather can change significantly within one hour, for which the best example is hail that has huge negative consequences in agriculture, and the second is that it is extremely time consuming. It is necessary to check the website or application every hour in order to obtain the latest data. This problem needs to be solved in a different way, that is, instead of the user checking the data himself and informing himself in that way, why wouldn't it be the other way around?

Assuming the wide presence of Internet and Internet of Things (IoT) network of interconnected edge devices, it is possible to implement a notification system that works automatically and notifies the user, by a method that suits him - SMS, Email or API Callback (calling an external URL, described in detail in the next section), when certain conditions are met. Example - the user can choose to be notified via SMS when snow falls on a location, such as a field in the countryside. Each time the data is collected, the system will check whether the conditions are met and, if so, send an SMS to the user that it's snowing at a given location. The user can set up multiple notifications and choose from multiple conditions so this functionality can save users time and prove to be very effective.

Given that for the analysis of meteorological data described earlier, a system of regular retrieval of data from available sources must be created, the notification system would be easy to implement. The system would be activated when entering the retrieved data, it would compare the current weather data for a location with user notifications activated for it, and if they match, it would send a notification in the manner specified by the user. One of the biggest challenges in this case is the user experience, ie. notification management interface. One must enter a large amount of data with different interrelationships (Variable - Temperature, Rain, etc.; Comparison - higher / lower, true / false; Value, etc.), and the process must be easily understood and used.

After certain period and several attempts, a specific design and user experience solution was selected - Conversational User Interface [13]. The interface designed in this way is not an exact example of a conversational user interface, but it is modeled based on it and has several similarities. Its main feature is that notifications can be created and managed through a sentence-like interface whose parts can be changed and ultimately make perfect sense. This is achieved by combining textual elements and inputs with brief instructions designed to understand what to choose as that input.

Below are two types of sentences that can be used to create notifications (the underlined part of the text is input):

```
------------------------------------------------
Notify me when SELECT MEASUREMENT UNIT SELECT
COMPARISON ENTER VALUE at location, SELECT LOCATION,
notify me via SELECT METHOD on phone number /
email / callback URL _______________.

Notify me when SELECT PRECIPITATION at location,
SELECT LOCATION, notify me via SELECT METHOD on
phone number / email / callback URL _______________.
------------------------------------------------
```

By selecting the inputs in the given sentences, the following sentences can be obtained:

```
------------------------------------------------
Notify me when the temperature rises above 40 at
location Greenhouse, notify me via SMS at 069 123
456

Notify me when it rains at location Orchard, notify
me via Callback (URL) at callback URL
http://server.automation.control/device/1/activate.
------------------------------------------------
```

This method of notification management enables the best user experience and easy data management for end users, which is farmers in our case.

D. Machine/Equipment Automation

The functionality that shows a great potential is sending notifications via the Callback API. It builds on the notification system described in the previous section and is a method by which users can be notified in case the conditions for notification for a location are met. The Callback API in the world of programming is a way to get feedback to your software/hardware after something happens in an external IT system. In this case, for the end user, this information system represents an external system, and the programs and devices used by the user represent its software and hardware.

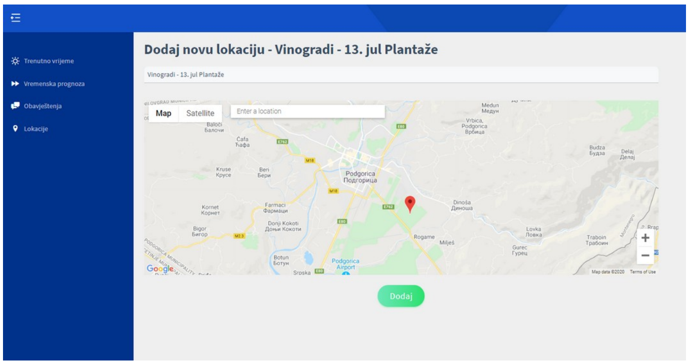
For example, user can set to receive a notification when hail starts to fall at one of its locations. The user may specify that he should be notified not via SMS or Email but via the Callback API, where it will provide an URL to its smart device server, which will, when it receives the notification, for example, automatically raise the cover to protect the plants from the hail. This method of communication with other systems is part of the IoT paradigm and opens up great possibilities for the automation of processes in agriculture [14, 15]. Irrigation control depending on air humidity and rainfall amount, raising and lowering the sun protection depending on temperature, activating anti frost systems, and much more can be automated, even systems that do not yet exist on the market, because this method allows a great possibility for interoperability with third-party and future systems.

E. Proof of Concept: A Prototype

All of the functionalities described in this paper were implemented as initial prototype system that served as a proof of concept [16]. The solution was implemented as a web-based service using Javascript programming language. Javascript was a language of choice, both on backend and frontend, with the exception of some processes that were developed with Python. The backend was done on Node.js, using the Strapi framework for developing data management system [17,18]. Frontend was developed using open-source framework Vue.js [19]. Finally, the data analysis and notifications system was developed purely in Python.

The system works as a web portal that is accessible via mobile device or personal computers. Registered users can specify several locations of interests and for each location weather data is collected, aggregated and processed as described in previous sections. Adding a new user defined location is illustrated in Fig. 4. The user can pick a location using a map based user interface. Each user can define several locations of interests and the data are kept and displayed in dedicated user space.

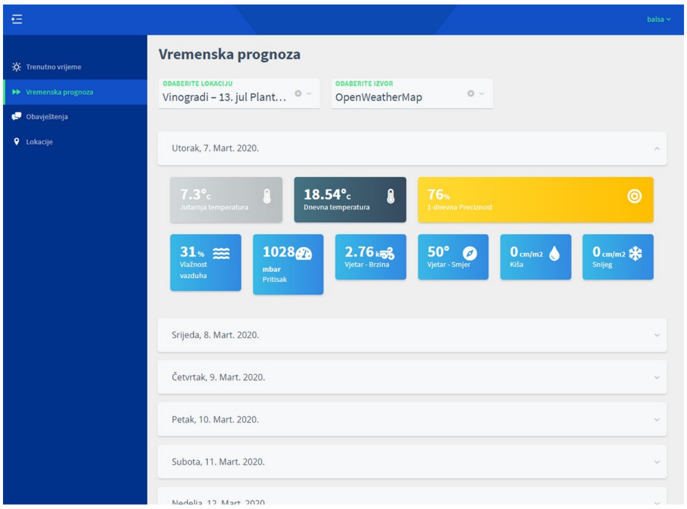
Once configured, the user access the data via user dashboard, where she or he can select the location of interest, data sources, type of data (current or forecast), etc. User dashboard of the prototype system is illustrated in Fig. 5. In addition to accessing data via dashboard, the user can specify notification/alarms conditions in order to be notified via SMS and/or email. Finally, the prototype allowed for communication with IoT edge actuators capable of receiving commands via URL using the Callback API.



Dodaj novu lokaciju - Vinogradi - 13. jul Plantaže

Vinogradi - 13. jul Plantaže

Fig. 4. Creating a user defined location for meteorological data aggregation and analytics - vineyard at 13. jul Plantaže, Podgorica, Montenegro



Vremenska prognoza

ODABERITE LOKACIJU: Vinogradi – 13. jul Plant...

ODABERITE IZVOR: OpenWeatherMap

Utorak, 7. Mart. 2020.

7.3°c Jutarnja temperatura

18.54°c Dnevna temperatura

76% 1-dnevna Preciznost

31% Vlažnost vazduha

1028 mbar Pritisak

2.76 km/h Vjetar - Brzina

50° Vjetar - Smjer

0 cm/m2 Kiša

0 cm/m2 Snijeg

Srijeda, 8. Mart. 2020.

Četvrtak, 9. Mart. 2020.

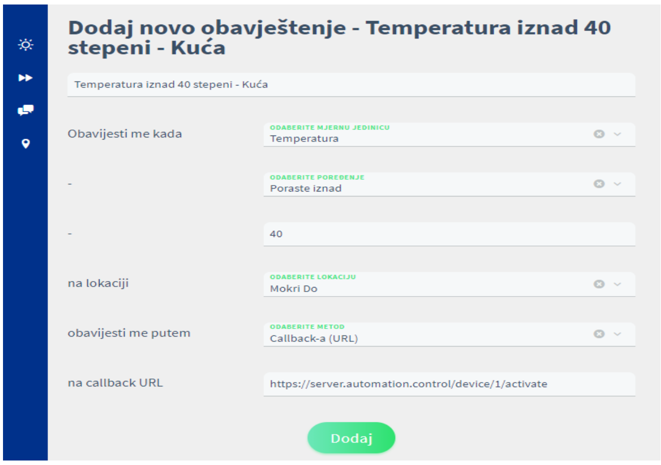
Petak, 10. Mart. 2020.

Subota, 11. Mart. 2020.

Nedelja, 12. Mart. 2020.

Fig. 5. User dashboard showing the weather forecast for the selected location

Configuring of an alarm for one of the selected locations is illustrated in Fig. 6. The user selects the location, provides a name for the alarm and defines conditions, i.e. notify me when the temperature rises over 40 degrees.



Dodaj novo obavještenje - Temperatura iznad 40 stepeni - Kuća

Temperatura iznad 40 stepeni - Kuća

Obavijesti me kada — ODABERITE MJERNU JEDINICU: Temperatura

- — ODABERITE POREĐENJE: Poraste iznad

- — 40

na lokaciji — ODABERITE LOKACIJU: Mokri Do

obavijesti me putem — ODABERITE METOD: Callback-a (URL)

na callback URL — https://server.automation.control/device/1/activate

Fig. 6. Configuring an alarm for high temperature for the selected location

IV. Conclusion

One of the decisive factors in agriculture is the weather and it's impact on crop growth and their yields. Having precise information on the expected amount of precipitation, farmers can optimize crop irrigation and thus increase yields while reducing costs, knowing wind speed in the coming period they can plan soil pollination, knowing temperature and humidity of the soil they can estimate the best time for fertilization. These examples illustrate the importance of predicting weather conditions for the success of farmers' yields.

This paper describes an information system whose functions are aimed at providing reliable meteorological data to farmers, improving the efficiency of their work through notifications and through the automation of their mechanization. In order to obtain the most accurate meteorological data, users have the option of creating micro locations, as well as creating condition based notifications, which are sent either to the user, or directly to his devices (via API callback) in case that conditions are met at a specific location defined by the user. For example, an increase in temperature above 40 degrees Celsius, or a decrease in humidity below 20% and the like.

The information system is designed in such a way that it can be developed and expanded in several ways. It is possible to add more data sources, add new analyzes and thus further "enrich" raw data, and it is possible to expand the system by adding new users' sensors and devices.

New features and protocols will be developed in the future, but certainly all the data from this information system, both raw and derived through the processing, can be exported for further analysis. The main purpose of the system is to provide timely and accurate meteorological data to farmers and the system can easily be interfaced and integrated with other tools from Agriculture 2.0 ecosystem. The system is open for adding new data analytics and future work may include the use of machine learning and creation of new decision support functions.

Acknowledgment

The research is supported by the scientific research project “Digital transformation of agriculture production and food supply chain in Montenegro - DIPOL”, though the grant awarded by the Ministry of Science of Montenegro, grant agreement No. 01-788/2.

References

[1] J.P. Hausberg et al., “Research streams on digital transformation from a holistic business perspective: a systematic literature review and citation network analysis,” Journal of Business Economics, 2019(89), 2019, pp 931–963.

[2] A. Kreitstshtein, “Digital transformation and its effects on the competency framework: a case study of digital banking,“ Bachelor Thesis, University of Applied Sciences, Haaga-Helia ammattikorkeakoulu, 2017.

[3] D. A. Skog, “The Dynamics of Digital Transformation : the Role of Digital Innovation, Ecosystems and Logics in Fundamental Organizational Change,” PhD dissertation, Umeå Universitet, 2019.

[4] V. Erceg, T. Zoranović, “Digital transformation of business in agriculture: redesign of business processes,“ Agroeconomica, 48(85),Faculty of Agriculture, Novi Sad (Serbia). Department of Agricultural Economics and Rural Sociology, 2019, pp 67-73

[5] B. Ozdogan, A. Gacar, H. Aktas, “Digital agriculture practices in the context of agriculture 4.0,“ Journal of Economics, Finance and Accounting, 2017(13), Pressacademia, 2017, pp 184-191

[6] M. Darr, “Big Data—The catalyst for a transformation to digital agriculture,“ Proceedings of the 26th Annual Integrated Crop Management Conference, 2014, pp 25-26

[7] A. Kaloxylos et al., “Farm management systems and the Future Internet era”, Computers and Electronics in Agriculture, Vol 89, 2012, pp 130-144.

[8] U. Deichmann, A. Goyal, D. Mishra, “Will digital technologies transform agriculture in developing countries?,” Agricultural Economics, 47(S1), 2016, pp 21-33.

[9] T. Popović et al, “Architecting an IoT-enabled platform for precision agriculture and ecological monitoring: A case study,” Computers and Electronics in Agriculture, Volume 140C, August 2017, pp 255-265.

[10] V. Maraš et al., “Precision Viticulture Using Wireless Sensor Network,” 2020 9th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2020, pp. 1-6.

[11] S. Damjanović et al., “Smart Specialization Strategy of Montenegro 2019-2024,“ Ministry of Science of Montenegro, 2019.

[12] K.G. Hubbard, N.J. Rosenberg, D.C. Nielsen, “Automated Weather Data Network for Agriculture,” Journal of Water Resources Planning and Management, 109(3), 1983.

[13] M.F. McTear, “Spoken Dialogue Technology: Toward the Conversational User Interface,” 1st Ed., Springer-Verlag, London, 2004.

[14] G.A.R Garcia-Perez et al., ”Autonomous robot in agriculture tasks,” In: 3ECPA-3 European Conference On Precision Agriculture, 2001, pp 25-30.

[15] Z. Nakutis et al, “Remote Agriculture Automation Using Wireless Link and IoT Gateway Infrastructure”, 2015 26th International Workshop on Database and Expert Systems Applications (DEXA), Valencia, 2015, pp. 99-103.

[16] B. Lazarević, „Management of meteorological data for agricultural applications,“ Master Thesis, Faculty for Information Systems and Technologies, University of Donja Gorica, Montenegro, June 2020.

[17] Node.js JavaScript Runtime. Accessed August 2020. [Online]. Available: https:/nodejs.org

[18] Strapi. Open source Node.js Headless CMS. Accessed August 2020. [Online]. Available: https://strapi.io

[19] Vue.js. The progressive JavaScript Framework. Accessed August 2020. [Online]. Available: https://vuejs.org

One solution of safety core in autonomous driving systems

Maksim Egelja, Marko Dragojević, Nikola Teslić, *Member, IEEE, Research Institute RT-RK,*

Nemanja Lukić, *Member, IEEE, Faculty of Technical Sciences, University of Novi Sad*

***Abstract*—Automotive industry now days needs one of the most sophisticated software, as that it brings m any new challenges to the world of engineering. Recently, autonomous driving became one of the biggest challenges of this field and it imposes high safety procedures.**

Work presented in this paper defines modern autonomous driving safety core which is responsible for detecting and controlling the autonomous system when safety critical event occurs either outside or inside the vehicle.

***Index terms*—Autonomous driving; ROS; safety core; driving state machine**

I. Introduction

Over the past years engineering focus was on providing driving assistance systems, now days autonomous driving became one of the most demanding software industry branch since engineers are facing with many complex problems, especially safety critical algorithms. Due to the fact that there are many sensors attached to the vehicle such as LiDAR (Light Detection and Ranging), GPS (Global Positioning System), full range and short range radars, multiple cameras there is a lot of data to collect and process in real time. Having this in mind engineers need special hardware units with very powerful processors, which are able to execute very complex algorithms[1]. Most components such as image processing, which are used for autonomous driving, are already solved as individual problems. Those components are highly important for this engineering field, but fusion of them still needs to be defined, in order to construct fully automated vehicle.

Assuming all facts mentioned above there is a lot of room for various malfunctions of either hardware or software. Here is where safety[2] part of autonomous driving system plays the role, it needs to prevent system of controlling vehicle when some of factors are incorrect.

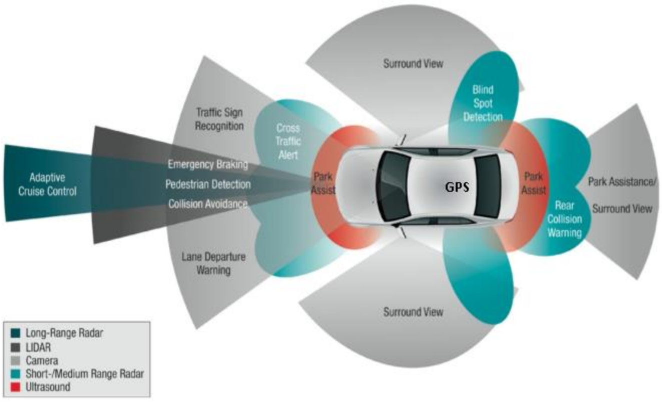


Fig. 1. Autonomous car with sensors.

Paper presents definition and implementation of autonomous driving system safety core.

The rest of material is organized as follows. At the beginning, section II describes software libraries and platform used in this research, while section III describes system architecture, where its sub sections present more detailed description of system. Section IV gives the experimental results of this work, whereas section V presents a conclusion of the demonstrated work and proposes some of its possible future improvements.

II. Platform and software libraries

Goal of this work was to define and make autonomous driving system safety core, capable of detecting and reacting to malfunctions of system's hardware or software components. AUTOSAR [3] is used as main development platform since it fulfills automotive safety standards. ROS [4] (Robot Operating System) and custom middleware were used as simulation and testing platform of this system. Today, ROS beside very high popularity in robot controlling software systems, finds usage in simulation and testing of autonomous driving systems. ROS specifies way of communication between nodes through specified topics. Nodes are completely independent processes and they are convenient to be used for simulating various parts of vehicle systems.

Safety core was realized using C programming language while the rest of work was realized using modern C++ programming language and tested using acceptance tests with fitnesse[5] framework, while units of this work are verified and tested using *gtest* [6] framework.

This work was partially supported by the Ministry of Science and Technological Development of Serbia under Gant III_044009_1 TR36029

Maksim Egelja, Marko Dragojević and Nikola Teslić are with the Research Institute RT-RK, 23a Narodnog fronta, 21000 Novi Sad, Serbia (e-mail: maksim.egelja@rt-rk.com, marko.dragojevic@rt-rk.com, nikola.teslic@rt-rk.com).

Nemanja Lukić is with the Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obardovića 6, 21000 Novi Sad, Serbia (e-mail: nemanja.lukic@rt-rk.com).

III. System Architecture

As explained before, ROS nodes were used as system units. Schematic view of realized safety core is shown in figure 2.

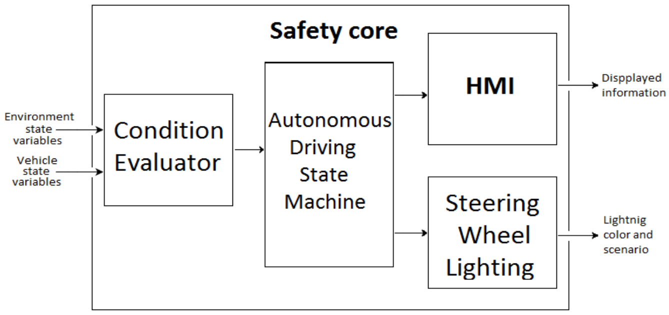


Fig. 2. Illustration of simulated hardware and signals between them.

Autonomous driving state machine is heart of the system. As its name says, this unit is in charge of automated driving system state. It switches states depends on various signals and internal parameters to keep vehicle and its passengers safe. Beside this unit there are condition evaluator, HMI(Human Machine Interface) and steering wheel lightning control units. Condition evaluator node analyses a lot of different signals such as door open signal, fasten seatbelt signal, etc. Once all signals are in right condition this part of system will allow activation of automated driving. Purpose of HMI and steering wheel lightning component is to inform the driver about autonomous driving system state and warn him about safety critical situations where human intervention is necessary.

All software components are implemented through AUTOSAR specific standard. Every software component is written using C programming language and has standardized architecture. Components are consist of receiver, core and sender part. Since AUTOSAR framework is used for implementation communication between components has to be done through particular memory zones through interfaces. Receiver part of component has to take all input interfaces from memory to components internal representations, then core part has to deal with component logic and sender part is there to write component output data back to communication memory. In further text more detailed explanation of each system unit is given.

A. Autonomous driving state machine

As previously mentioned autonomous driving state machine has to determine exact state of entire automated driving system.

It has several possible states:

- OFF the initial state, it changes when driver wants to activate autonomous vehicle pilot.
- READY state signify that all sensors and actuators as hardware units and all software components are ready to take over control of the vehicle. Also that means that all other conditions such as weather and traffic conditions are fulfilled.
- ACTIVE state represents fully automated vehicle driving. When system is in this state vehicle is capable of driving at certain speed and road conditions without human interaction.
- ACTIVE TOR (Take Over Request) state occurs when automated driving system detects situation which is impossible to handle by itself. In this state system notifies that human intervention is needed in certain period of time. There are three TOR levels, low, medium and high priority take over requests. Each priority level represents amount of time until system activates minimal risk maneuver.
- ACTIVE MRM (Minimal Risk Maneuver) is state of autonomous driving state machine which is entered when TOR timer e lapses without driver intervention. In this state system will guide the vehicle to complete safe stop. If driver take over control of vehicle during active TOR state, active MRM state will not be triggered.

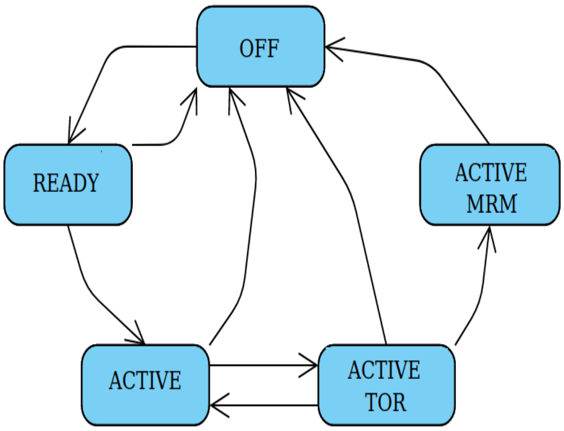


Fig. 3. Autonomous driving state machine state diagram.

As Autonomous driving state machine is FSM(Finite State Machine) it has its state transfer functions. Figure 3 shows its state transfer diagram using these functions. State transfer functions defines to which state this component under certain conditions will transfer from the current one. In further text description of state transfer is given.

From the off state system goes to ready when user wants to activate automated pilot only if all hardware and software units are working properly. If something goes wrong system will be set back to off state or everything went before goes to active state system will check all road and weather conditions. When vehicle figure out road situation or some of sensor reports an error state system will switch to active tor state where driver is informed about the problem and amount of time he has until minimal risk maneuver is performed. If driver takes over the control until tor time period and handles potentially risky situation, vehicle will continue to drive by itself, system will be back to active state. From all above mentioned states system can go back to off if user wants to switch of automated driving pilot.

B. Condition evaluator

This software component has to observe a large amount of signals and make sure that neither hardware or software components are working properly in context of automate vehicle pilot. Condition evaluator has to check some very basic signals such as are all doors on vehicle closed or are all passengers are fasten seat belt beside these there are some very complex signals to analyze such as weather conditions observed through camera or driver fatigue. All of signal checking hast to be done in real time. Having this in mind we had to provide communication interface fast enough and a lot of computing power and optimizations. Due to these facts we chose ROS as our communication platform and classic AUTOSAR as implementation platform.

Weather and driver fatigue signals analysis is done through multiple cameras. Before signal from camera even comes to condition evaluator a lot of image processing has to be done how could we make filtration and use only the most important data.

After all checking finishes this software components sends its output to autonomous driving state machine, it contains of state machine state to be allowed and TOR time if some of the signals to be checked is in error state.

C. *Human Machine Interface*

Now days we have modern, digital HMIs present on the automotive market. As all digital systems it provides as a huge amount of possibilities to be shown. In figure 4 illustration of modern human machine interface is given.

Fig. 4. Illustration of HMI used in research.

Beside its regular functions, to show vehicle speed and engine RPM(Rotation Per Minute) HMI in this system has few more functionalities. It has to inform driver about vehicle state in automated diving mode. HMI shows when automated driving mode is activated and if preciously mentioned TOR occurs lets the driver know. This software component is in direct communication with autonomous driving state machine as a heart of a system. It is highly important in this system because provides information about autonomous driving system to driver which is crucial for safety.

D. *Steering wheel lightning*

Today we are witnesses of LED(Light Emitting Diode) lightning era. As a modern system autonomous car has large amount of LEDs built in steering wheel. These LEDs have safety critical information function. Figure 5 shows the concept of LED steering wheel.

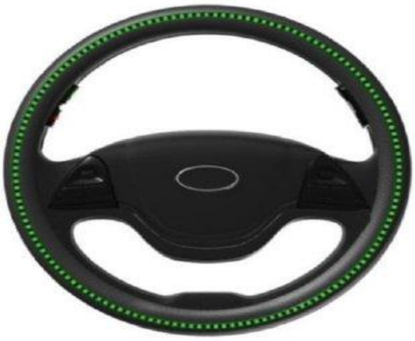


Fig. 5. Illustration steering wheel when autonomous pilot activated.

When system is activated and vehicle is driving on its own steering wheel is emitting green light, when some TOR occurs depending on its priority steering wheel is changing its color to blue, for low priority TORs, yellow, for medium priority and red, for the most urgent situations. Also red color is used when MRM is triggered.

Beside colors this component also changes lightning scenario based on TOR priority:

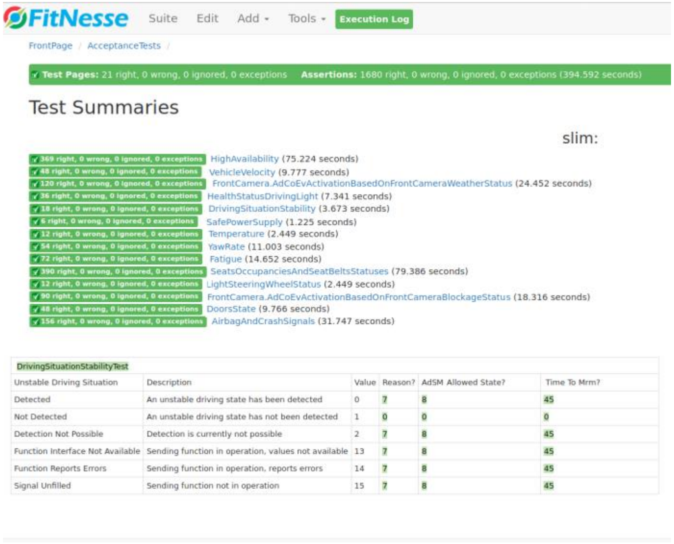
- Solid light - used when system is active.
- Pulsing light - used when medium priority TOR occurs.
- Blinking light - turns on when high priority TOR has been triggered.

Since modern LEDs are very powerful source of light and vehicle can be driven in very bright or extremely dark environment, dimming has to be performed. This is done by reading multiple brightness sensors which measures vehicles environment lightning and establishing minimal amount of light visible to human eye under current lightning conditions, since driver who is responsible for taking over control of the vehicle can always notice the safety warning and at the same time not be blinded by too high steering wheel illumination.

IV. Results

Since this research is realized in simulated environment and ROS is used as communication platform challenge was simulation and acceptance testing. All previously described software components needs to have wrapper implemented which is used to extract input and output data since software components are implemented according to AUTOSAR standard. Beside extraction these wrappers are used to transfer data over the ROS topics and instantiate ROS communication nodes for each of the software components.

Acceptance tests are performed using fitnesse open source framework. Tests are simulating various hardware or software malfunctions and verifies system behavior under given set of input parameters. In figure 6 results of acceptance tests is shown.



✓ **Test Pages:** 21 right, 0 wrong, 0 ignored, 0 exceptions **Assertions:** 1680 right, 0 wrong, 0 ignored, 0 exceptions (394.592 seconds)

Test Summaries

slim:

```
✓ 369 right, 0 wrong, 0 ignored, 0 exceptions  HighAvailability (75.224 seconds)
✓ 48 right, 0 wrong, 0 ignored, 0 exceptions  VehicleVelocity (9.777 seconds)
✓ 120 right, 0 wrong, 0 ignored, 0 exceptions  FrontCamera.AdCoEvActivationBasedOnFrontCameraWeatherStatus (24.452 seconds)
✓ 36 right, 0 wrong, 0 ignored, 0 exceptions  HealthStatusDrivingLight (7.341 seconds)
✓ 18 right, 0 wrong, 0 ignored, 0 exceptions  DrivingSituationStability (3.673 seconds)
✓ 6 right, 0 wrong, 0 ignored, 0 exceptions  SafePowerSupply (1.225 seconds)
✓ 12 right, 0 wrong, 0 ignored, 0 exceptions  Temperature (2.449 seconds)
✓ 54 right, 0 wrong, 0 ignored, 0 exceptions  YawRate (11.003 seconds)
✓ 72 right, 0 wrong, 0 ignored, 0 exceptions  Fatigue (14.652 seconds)
✓ 390 right, 0 wrong, 0 ignored, 0 exceptions  SeatsOccupanciesAndSeatBeltsStatuses (79.386 seconds)
✓ 12 right, 0 wrong, 0 ignored, 0 exceptions  LightSteeringWheelStatus (2.449 seconds)
✓ 90 right, 0 wrong, 0 ignored, 0 exceptions  FrontCamera.AdCoEvActivationBasedOnFrontCameraBlockageStatus (18.316 seconds)
✓ 48 right, 0 wrong, 0 ignored, 0 exceptions  DoorsState (9.766 seconds)
✓ 156 right, 0 wrong, 0 ignored, 0 exceptions  AirbagAndCrashSignals (31.747 seconds)
```

DrivingSituationStabilityTest					
Unstable Driving Situation	Description	Value	Reason?	AdSM Allowed State?	Time To Mrm?
Detected	An unstable driving state has been detected	0	7	8	45
Not Detected	An unstable driving state has not been detected	1	0	0	0
Detection Not Possible	Detection is currently not possible	2	7	8	45
Function Interface Not Available	Sending function in operation, values not available	13	7	8	45
Function Reports Errors	Sending function in operation, reports errors	14	7	8	45
Signal Unfilled	Sending function not in operation	15	7	8	45

Fig. 6. Acceptance test results.

These tests are testing system form high level, they simulates vehicle systems malfunctions and sends that simulated signals to condition evaluator which has to process the data and forward it to autonomous driving state machine which puts system in appropriate state depending on failure level. When state is determined, it is sent to steering wheel LEDs and human machine interface in order to inform the driver. Acceptance tests are checking all the data flow from simulated failure to information which driver receives through system components and make sure that safety core is performing without any mistakes.

Integration and unit testing is performed with usage *google test* and *google mock* library. Unit tests are verified software components functionalities while integration tests are verified connection between components and system functionalities all together.

V. Conclusion

Contribution of this work is successful implementation of safety core which can be used in highly automated vehicle pilots.

Investigation shows how one way of automated driving safety core implementation. Demonstration includes definition and development of software components which are mandatory for safety critical events which occurs while automated vehicle pilot is active.

Demonstrated safety core include following functionalities:

- Detection of hardware or software malfunctions using various health checking methods such as watchdog timers, direct health state signal checking, etc.
- Detection of complex traffic situation or bad weather conditions where sensors attached to vehicle cannot work properly.
- Informing the diver about detected situations and requesting take over in certain period of time.
- Performing minimal risk maneuver if driver does not take over control of the vehicle and led it to complete safe stop.

Since this research shows solution of safety core implementation in simulated environment, one of possible future improvements could be adapting the system to run on specialized hardware. Also more sensors and signals can be added to the system to make it more complex. As ROS is used as base communication platform of the system, it can be adapted to work on AUTOSAR Adaptive[7] platform which is more often used in modern vehicles software solutions.

References

[1] J. Levinson , J. Askeland , J. Becker , J. Dolson , D. Held , S. Kammel , J. Z. Kolter , D. Langer , O. Pink , V. Pratt , M. Sokolsky , S. Ganymed , D. Stavens , A. Teichman , M. Werling , S. Thrun "Towards fully autonomous driving: Systems and algorithms", IV, pp.163-168 , Baden-Baden, Germany, June 2011.

[2] L. Daxue, A. Xiangjing, S. Zhenping, H. Hangen "Active Safety in Autonomous Land Vehicle", PEITS, pp. 476 - 480, Guangzhou, China, August 2008.

[3] AUTOSAR Technical overview, 2007, AUTOSAR Specification Release 3.0, Retrieved on 28/11/2007.

[4] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng "Ros: an open-source robot operating system", ICRA Workshop on Open Source Software, vol. 3, no. 3.2, pp. 5, June 2009

[5] G.K. Hanssen, and B. Haugset, Automated Acceptance Testing Using Fit, in 42d Hawaiian International 20 Conference on System Sciences (HISS'09). 2009, IEEE Computer Society: Hawaii, USA. P.1-8.

[6] J. Swaminathan "Test-Driven Development" in "Mastering C++ Programming", Birmingham, United Kingdom, Packt Publishing, September 2017

[7] S. Fürst and M. Bechter, "AUTOSAR for Connected and Autonomous Vehicles: The AUTOSAR Adaptive Platform", 2016 46th Annual IEEE/IFIP Int. Con. on Dependable Systems and Networks Workshop (DSN-W), 2016.

One solution of keep lane function and curvature calculation with SD map data using ROS

Maksim Egelja, Stevan Stević, Nikola Teslić, Member, IEEE, Research Institute RT-RK,

Nemanja Lukić, Member, IEEE, Faculty of Technical Sciences, University of Novi Sad

***Abstract*—Software in automotive industry today is a very popular branch and brings many new challenges to the world of engineering. Recently, autonomous driving became one of the biggest challenges of this field. Goal is to develop a system capable of controlling vehicle almost completely independently.**

Work presented in this paper defines modern autonomous driving algorithm, which includes functionality of keeping autonomous vehicle in appropriate lane using SD (Standard Definition) map data instead of HD (High Definition). This algorithm is based on data fusion from camera senor and map data.

***Index terms*—Autonomous driving; ROS; vehicle control based on camera and map; keep lane function; curvature calculation**

I. Introduction

Recently, autonomous driving became one of the most popular software industry branches since engineers are working on providing highly automated driving systems, while over the past few years more focus is given on providing driving assistance. Due to the fact that there are many sensors attached to the vehicle such as GPS (Global Positioning System), LiDAR (Light Detection and Ranging), full range and short range radars, multiple camera, ultra sonic sensors[1], there is a lot of data to collect and process in real time. Having this in mind engineers need huge computing power, special hardware units with very powerful processors, which are able to execute very complex algorithms. Sensors and their field of operation are illustrated in figure 1.

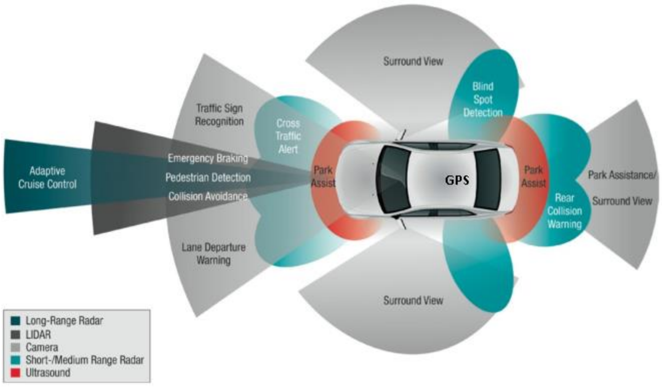


Fig. 1. Autonomous car with sensors.

Most components such as image processing[2], object detection and tracking[3], which are used for autonomous driving, are already solved as individual problems. Those components are highly important for this engineering field, but fusion of them still needs to be defined, in order to construct fully automated vehicle. In order to achieve this goal, engineers need to combine a great deal of research from different areas to produce highly automated vehicle pilot. Paper presents definition and implementation of road curvature calculation using fusion of camera sensor and SD instead of HD map data.

The rest of material is organized as follows. At the beginning, section II describes software libraries and platform used in this research, while section III describes system architecture, where its sub sections present more detailed description of system. Section IV gives the experimental results of this work, whereas section V presents a conclusion of the demonstrated work and proposes some of its possible future improvements.

II. Platform and software libraries

Goal of this work was to make autonomous vehicle system capable of controlling vehicle on roads where HD maps are absent. ROS [4] (Robot Operating System) was used as a platform of this system. Today, ROS is a very popular software system used for robot control, where the self driving vehicle can be defined as robot. Units of a system are implemented as ROS nodes where nodes are processes that

This work was partially supported by the Ministry of Science and Technological Development of Serbia under the project No. TR36029, year 2019

Maksim Egelja, Stevan Stević and Nikola Teslić are with the Research Institute RT-RK, 23a Narodnog fronta, 21000 Novi Sad, Serbia (e-mail: maksim.egelja@rt-rk.com, stevan.stevic@rt-rk.com, nikola.teslic@rt-rk.com).

Nemanja Lukić is with the Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obardovića 6, 21000 Novi Sad, Serbia (e-mail: nemanja.lukic@rt-rk.com, dragan.samardyija@rt-rk.uns.ac.rs).

are completely independent and can communicate with each other through specified topics.

Complete work was realized using modern C++ programming language and tested using acceptance tests with simulator and fitnesse[5] framework, while units of this work are verified and tested using *gtest* [6] framework.

III. SYSTEM ARCHITECTURE

As explained before, ROS nodes were used as system units. Schematic view of the system is shown in figure 2.

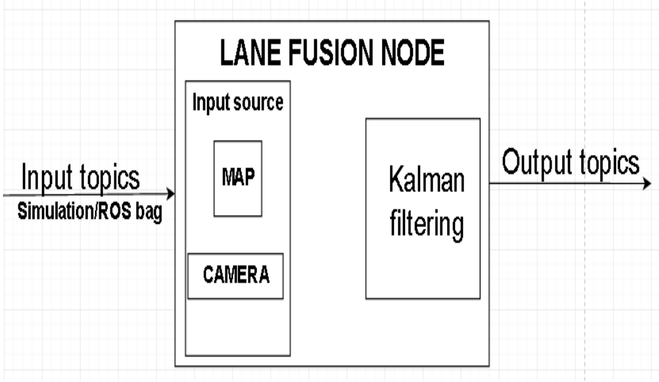


Fig. 2. Illustration of simulated hardware and signals between them.

Lane fusion node is main part of this autonomous vehicle subsystem. As its name says, this node is in charge of fusion data collected from camera and maps, it works very efficient when using HD maps, goal of this research is to make similar efficiency while using SD maps with less data available. Besides Lane fusion there are other nodes present, they are mainly used for providing data to lane fusion node. Since verification of algorithm presented in this work is done using acceptance tests, there are nodes which are in charge of simulation framework. In further text, more detailed explanation of system and HD maps is given.

A. HD map

Todays, well known, maps which are used in navigation devices and mobile phones are primary meant for humans, due to the fact that they can understand simple visual or vocal instructions provided from various devices. In autonomous diving era, where machines have to make decisions on the road, need for new set of maps is occurred. These maps are specially built for robotic systems, specifically they have very high precision, at centimeter level. This high precision is needed because self-driving car needs extremely precise instructions how to maneuver around 3D (3 Dimension) space. Tolerance for error is high in most of every day cases but there are a lot of situations where car has no room for error such as driving on the road build next to the cliff. Maps should contain where the lanes are, what is position of all traffic signalization, where the road boundaries are, maps also must contain data such as where the curves are and how high the curves are. When it comes to a self-driving car to precisely locate itself, lot of sensors are involved, visual sensors have a lot of limitations. Here the LiDAR sensor plays a role it precisely measures the depth or distance in a 3D space with lasers. However, both LiDAR and cameras as visual sensors, typically work together, running very fast, multiple times per second.

B. HD maps layers

An HD Map is organized into five layers. They are a base map (standard definition map), geometric map, the semantic map, map priors, and real-time knowledge as shown on figure 3.

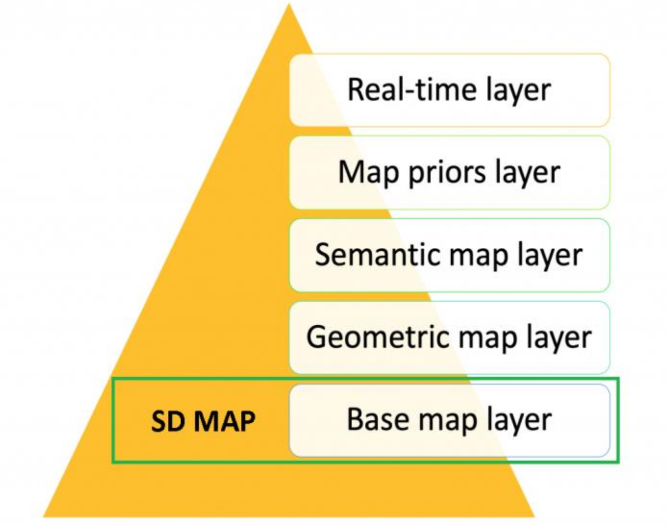


Fig. 3. Illustration of HD map layers.

Geometric Map layer contains of raw data collected by various visual sensors such as LiDAR and camera. Data is reconstructed and stored in this map layer. This layer contains information about positions of static objects near the road, e.g. buildings.

Semantic Map layer is geometrics layer upgrade were semantic objects are added. Semantic objects can be either 2D or 3D objects such as intersections, lane boundaries, parking places, traffic signalization, etc. These objects contain information about lane change restrictions, one way roads, usual traffic speeds, etc.

Map priors layer contains dynamic information such as average waiting time on traffic lights, probability of finding parking spot, etc. Autonomy algorithms commonly use this data in models as inputs and combines them with other real-time information.

Real-time knowledge layer is dynamically updated contains real-time traffic information. This data can also be shared in real time between autonomous vehicles.

C. Kalman filer

Kalman filtering[7], also known as LQE (linear quadratic estimation) is an algorithm that uses measurements observed over the time, such as from sensors, containing raw senor data with statistical noise and other interference. This algorithm produces estimates of unknown variables by estimating a probability distribution over variables. These estimates are more accurate than those based on a single measurement alone. As such, it is a common sensor fusion and data fusion algorithm.

The Kalman filter has numerous applications in engineering, mostly for guidance and navigation. These filters also are one of the main topics in the field of robotic motion planning and control, and they are included in trajectory optimization.

The algorithm works in a two-step process. In first, prediction step, the Kalman filter produces current system state variables estimates, along with their uncertainties. When next measurement arrives, usually corrupted with random noise and some other errors, estimates are updated using a weighted average method, which means that more weight will be given to estimates with higher certainly. Algorithm is recursive and can be run in real time, using only the present input measurements and the previously calculated state with its uncertainty matrix. There are extensions and generalizations of the algorithm developed, such as extended Kalman filter and unscented Kalman Filter which are used for nonlinear systems.

D. Lane fusion node

The lane fusion node is component in the perception module of autonomous driving stack, responsible for providing fused lanes output, which is the internal representation of lanes, lane geometry and lane attributes. Lane fusion fuses sensor and map data according to provide fused lanes output shown on the figure 4.

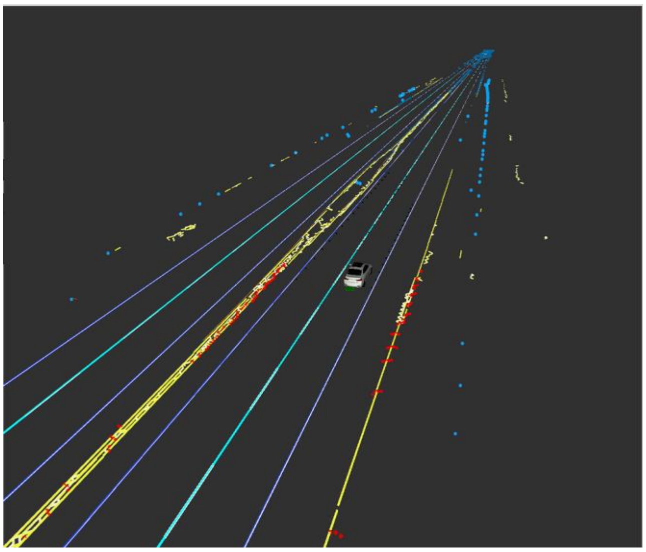


Fig. 4. Graphical view of fused lanes.

This node is using Kalman filtering algorithm for making predictions if sensor data is not very reliable, e.g. camera lens is dirty or there are bad weather conditions, fog, sun shining directly to camera sensor and etc.

Since we are using ROS as our development platform lane fusion component is distributed on two layers, the communication layer, handles all information exchange through specified ROS topics and the domain layer containing sensor and map data processing units. Figure 5 shows architecture diagram of lane fusion component.

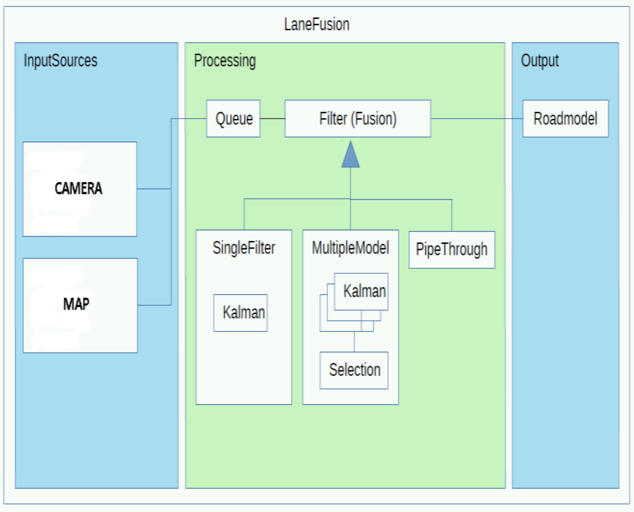


Fig. 5. Lane fusion software architecture diagram.

Communication layers main class, LaneFusionApplication, is responsible for communication with other autonomous driving system components. It includes communication interface objects used for publishing and subscribing messages.

Domain layers main class, LaneFusion, is the entry point in this layer. It comprises objects for each input source and a fusion algorithm called FusionStrategy. This part of a system is triggered cyclically by the LaneFusionApplication. When triggered, it sets up a queue of observations and gives it to fusion algorithm.

E. Map source

Map source is subcomponent of lane fusion node, it is in charge of providing map data to fusion algorithm. Also it calculates road curvature using data available.

HD map data curvature calculation is done by observing center lines from each lane on the road while curvature calculation based on SD map data is done using only one center line per road. All curvature calculations are done by using *intel AdLib* library, part of code where curvature calculation performs is attached in listing 1.

```
AdLibCurveFitting<Point2d> curve_fitting(polyline);

if (curve_fitting.GetStatus() == 0)
{
  auto curvature =
curve_fitting.CalculateCurvatureAt(arc_length.value());

  if (curvature.has_value())
  {
    return curvature.value();
  }
}
```

Listing 1 Curvature calculation C++ code

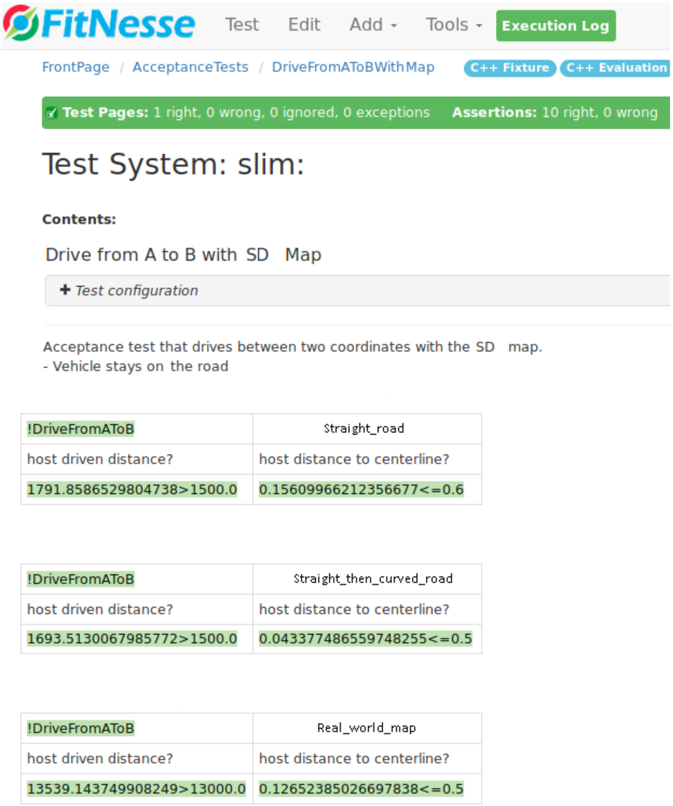
IV. Results

Testing of system is done in multiple ways, firstly acceptance tests are performed to check lateral displacement of the autonomous vehicle.

Tests are executed multiple times and in different scenarios:

- Straight road only
- Straight then curved road
- Straight then curved then straight road
- Curved road only with different curve radius
- Combined scenario with real world road example

All acceptance tests are done using fitnesse open source framework by checking vehicle driven distance and lateral vehicle displacement, results of combined test scenario are shown in figure 6.



✓ **Test Pages:** 1 right, 0 wrong, 0 ignored, 0 exceptions **Assertions:** 10 right, 0 wrong

Test System: slim:

Contents:

Drive from A to B with SD Map

+ *Test configuration*

Acceptance test that drives between two coordinates with the SD map.
- Vehicle stays on the road

!DriveFromAToB	Straight_road
host driven distance?	host distance to centerline?
1791.8586529804738>1500.0	0.15609966212356677<=0.6

!DriveFromAToB	Straight_then_curved_road
host driven distance?	host distance to centerline?
1693.5130067985772>1500.0	0.043377486559748255<=0.5

!DriveFromAToB	Real_world_map
host driven distance?	host distance to centerline?
13539.143749908249>13000.0	0.12652385026697838<=0.5

Fig. 6. Acceptance test results.

Unit and integration testing is done in this research too. It is done by using *google test* and *google mock* library. Mocking the objects is designed with use of DI (Dependency Injection) design pattern.

Further data analysis is done in this research too, curvature calculated using SD map is compared to values obtained from HD map. In the upper diagram of following figure 7 graphical view of comparison curvature values over the time is given, while the lower diagram shows curvature comparison with its standard deviation.

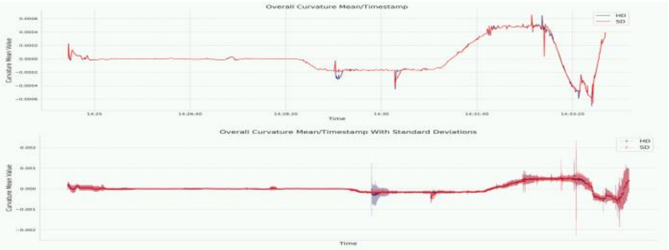


Fig. 7. Output of testing script used to check system functionality.

V. Conclusion

Contribution of this work is successful implementation of usage SD instead of HD map data in self-driving cars.

This research shows how fusion of SD map and camera data can be done in autonomous driving system. Demonstration includes definition and development of algorithms, based on data fusion, for vehicle control using camera and SD map data. Algorithms shown in this investigation are:

- Keep lane function where software components are keeping a vehicle in current lane.
- Curvature calculation where software units calculate curvature of the road in front of vehicle using SD map data which consist of much smaller amount of information compared to HD map.

The fact that embedded systems like self-driving car, has a limited amount of memory, usage of SD data is very useful, also all parts of the world will not have HD maps available, so system must rely on only available SD map data. By selection of ROS as the base platform of this work, program code is platform independent. Next step could be transferring the whole system to other platforms such as *AUTOSAR*[8] framework.

References

[1] H. Somogyi, D. Pup, P. Körös, A. Mihaly, A. Soumelidis "Research of Required Vehicle System Parameters and Sensor Systems for Autonomous Vehicle Control", 12th SACI, Timisoara, Romania, pp. 27-32, May 2018

[2] H. Kurihata "Rainy weather recognition from in-vehicle camera images for driver assistance", IEEE Proceedings. Intelligent Vehicles Symposium, Las Vegas, NV, USA, pp. 205-210, June 2005.

[3] T. Ogawa, H. Sakai, Y. Suzuki, K. Takagi and K. Morikawa, "Pedestrian detection and tracking using in-vehicle lidar for automotive application", Baden-Baden, Germany, pp. 734-739, June 2011.

[4] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng "Ros: an open-source robot operating system", ICRA Workshop on Open Source Software, vol. 3, no. 3.2, pp. 5, June 2009

[5] G.K. Hanssen and B. Haugset, Automated Acceptance Testing Using Fit, in 42d Hawaiian International 20 Conference on System Sciences (HISS'09). 2009, IEEE Computer Society: Hawaii, USA. P.1-8.

[6] J. Swaminathan "Test-Driven Development" in "Mastering C++ Programming", Birmingham, United Kingdom, Packt Publishing, September 2017

[7] J. Xi, Kalman filter and its application foundation[M], National Defense Industry Press, 2001.

[8] AUTOSAR Technical overview, 2007, AUTOSAR Specification Release 3.0, Retrieved on 28/11/2007.

Modern JavaScript 3D Visualization Frameworks: A Comparative Study

1st Vladislav Tokovic
Technical Faculty
Singidunum University
Belgrade, Serbia
vladislav.tokovic.19@singimail.rs

2nd Miodrag Zivkovic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
mzivkovic@singidunum.ac.rs

3rd Nikola Savanovic
Faculty of Informatics and Computing
Singidunum University
Belgrade, Serbia
nsavanovic@singidunum.ac.rs

***Abstract*—Web applications are the most popular type of modern applications. The trigger, which increased the popularity of the web applications, was the interactivity, provided with the introduction of the JavaScript language. Using JavaScript web sites became dynamic, with the possibility to communicate with site users. JavaScript today is not only limited to the client side, but can also be used as the server side language. On the other side, modern browsers have also improved in the recent time. Amongst others, one exiting capability, which can be visually appealing to the end users, is the possibility to render interactive 3D graphics within the browser itself. The focus of this research is to give an overview of the popular 3D graphics JavaScript frameworks: Babylon.js, ClayGL and Three.js. Within this paper, we have compared the frameworks, and presented the advantages and disadvantages of each framework.**

Keywords—JavaScript, client applications, 3D graphics, animation, Babylon.js, ClayGL, Three.js

I. Introduction

Traditional applications were implemented as desktop solutions. However, in the recent times, the majority of applications shifted to the web or mobile platforms. The potential of web applications was recognized almost 20 years ago. In one of the early research papers on the topic [1], it was noted that the major drawback of the desktop applications is the need to distribute them and install them locally, and that the web applications were very promising and could overcome them. However, the main drawback of the web applications of the time was the very limited interactivity. The interactivity was added to the web applications with the introduction of the client side script language JavaScript, which enabled building responsive web sites. However, the look and feel of traditional desktop applications was still out of the reach, mostly because of the need to contact the web server with HTTP request for every new content needed to be displayed on the web page. The true interactivity, which triggered the massive shift from traditional desktop applications to the web and initiated the web revolution, was the introduction of Ajax [2]. Ajax is approach that relies heavily on JavaScript, combined with several other technologies, such as HTML, DOM and XML.

The modern browsers offer new capabilities, with support for rendering and displaying 3D graphics. This particular capability is bringing modern web sites and the look and feel that they offer to the clients to the completely new level. The users of the web application will be attracted to the novel design and content presentation, and therefore, increase the popularity of the given web application. The goal of this paper is to present three popular JavaScript frameworks which deal with 3D animations: Babylon.js, ClayGL and Three.js. We tested all frameworks on the same animation examples, in order to show advantages and disadvantages of each framework, and to show side-by-side analysis and comparison.

The rest of the paper is organized as follows. Chapter two gives a short description of each of the observed frameworks. Chapter three deals with the assessment of each framework performance in the context of specific animations. Chapter four gives the comparative analysis of the observed frameworks. Finally, chapter five gives final observations and concludes the paper.

II. Popular JavaScript 3D Graphics Frameworks

The browser capability to display 3D graphics comes from the WebGL, which allows the browsers to access the graphic cards for additional processing power [3]. WebGL utilize GLSL ES for communication with the graphic cards, as it requires different instructions set than common languages needed for the traditional web application development [4]. Traditional web developers are not familiar with this syntax. In effort to simplify the process of developing animations, there are numerous JavaScript libraries which take care of the communication with the WebGL. Quick Google search ends up with more than 20 different available frameworks for JavaScript animation. It is hard to compare the pros and cons of all these libraries in order to find the most suitable WebGL framework for the specific need. This paper investigates the most popular frameworks in order to help developers choose the most suitable framework for their purposes. In this chapter, we provide the short overview of three most popular frameworks which we have analyzed.

A. Babylon.js

Babylon.js is a JavaScript framework that was designed for creating 3D applications and video games for Web [5]. Currently, active version of Babylon.js is 4.1, as of 2020 [6]. It was published as an open source tool in 2014 [7]. It has a community of users which continues to grow every day and which actively contributes to the projects by improving the existing and adding new features.

Babylon.js has been developed by using a TypeScript, which is a language that is used to generate pure JavaScript files. Within the framework, all tools required for creating 3D applications are embedded. It supports loading and drawing 3D objects, manipulating them, adding special effects. As it was designed as a tool for developing web games as well, it supports adding sounds and creating gameplays as well. According to the authors [6], it is an easy-to-use tool which can be set up with the minimum lines of code.

Babylon.js has also several very useful online tools, which can help both developers and artists to get familiar with the framework and test their creations. Babylon Playground is a live code editor, which can be used for experimentation and training. It displays a live editor with support for autocompletion, together with the canvas to see the effects right away. Another tool is the Sandbox, which supports drag and dropping already exported scenes directly to the web browser to see the results practically in real time.

B. *ClayGL*

ClayGL is another Web 3D graphics framework, used for building 3D web applications. It is an open source project, with the currently active version 1.3.0 [8].

It is a relatively young framework, but it is worth mentioning. When only using the core functionality of the framework, it is as small as ~25 KB, because of its modularity. ClayGL is getting on the popularity, however, as it still doesn't have a large community of users, the available documentation is scarce to say at least. On the other hand, there is a rich set of examples available to help the kick start with this framework. Additional features worth mentioning include scaled graphics, zooming support and highly detailed graphics with numerous advanced and very detailed materials.

C. *Three.js*

Three.js was created in 2010, and it has been available on the GitHub ever since [9]. Over the past decade, the framework has grown a large community of users, many of them contributing to the framework with new functionalities [10]. As it has been around for a while, and being the most famous 3D JavaScript framework, it has a large community of thousands of developers. This framework has great support, as well as community support. Also, some studies says that in real production *Three.js* is faster than *Babylon.js* framework [12] .

Three.js can be used for 3D animations for web sites, but it is also popular for WebGL-based video games and simulations. It is considered to have easy learning curve, and since it is around for so long, it has hundreds of examples and tutorials, and a huge and active community. It also provides a powerful, yet lightweight online editor.

It has been used in numerous video games, and also in many scientific and medical visualizations. For example, Miew: 3D Molecular Viewer, which is a web tool for advanced visualization and manipulation of molecular structures, and which is used in various fields of medicine, chemistry, biology etc. has been developed using Three.js [11]. Authors chose these three frameworks for comparison because all of these WebGL frameworks are widely used, and authors wanted to represent similar animations in three different environments to show their performance.

III. Assessment of Framework performances in the context of specific animations

To correctly evaluate the quality of animations developed in these JavaScript frameworks, we have created two types of animations. The idea is that analysis of frameworks of interest (that we discuss in this paper) will be more adequate if it is performed in the context of visually and functionally similar scenes. As the result of this approach, we were able to gain insight to framework possibilities first-hand, from the web designer's perspective, by building our animation and doing problem solving and optimization in real-time, with developing resources limited to one framework. Afterwards, we were able to inspect similar animations side-to-side and have certain amount of confidence that visual differences among animations stemmed from differences in frameworks, which we would then focus on.

The first animation (Fig. 1.) consists of 7 shapes. The central position in the scene is occupied by yellow sphere placed partially inside three yellow cubes rotating around y-axis (vertical axis). There is one rotating cube to the left and one to the right of the central object, and one gray sphere, which is the only scene element that is changing its position at runtime – following the vertical circular path around the central piece. Aside from these elements, there are 4 directional light sources present in the scene.

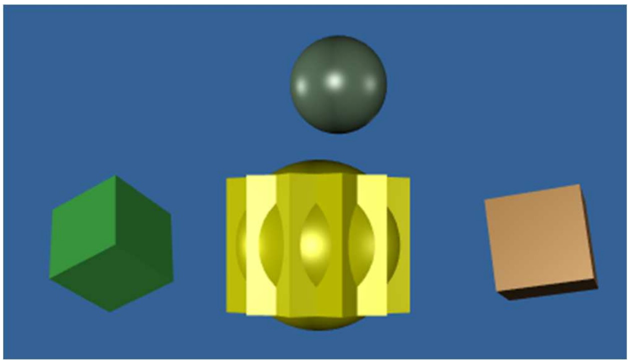


Fig. 1. First animation visual representation

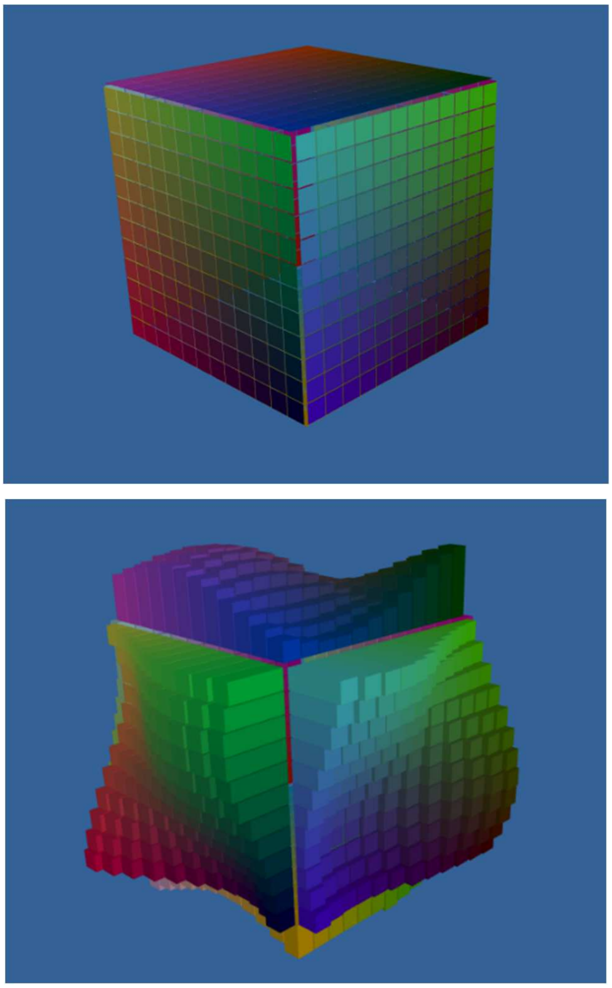


Fig. 2. Second animation

The second animation (Fig. 2.) consists of one large cube with edge length of 12 units and three directional light sources. Each of 6 sides of the cube is subdivided into boxes of the same length and width, spaced with small uniform distance between each other (Fig. 2. top) and colored to form a gradient. For the purpose of evaluating the graphic rendering capabilities of each framework, we will provide some relevant metrics. Edge length of large cube is 12 units, area of each side is 12^2 = 144 squared units and total area of the large cube is 6*144=864 squared units. Another important quantity would be the area of the large cube that is fully visible from the client perspective, which, if the camera is position as in Fig. 2., is half the area of the large cube (3 out of 6 sides are visible), which is 864/2=432 square units. At runtime, three aspects of the animation can be manipulated:

- Cube density

The number of boxes per square unit can be increased. In the example on Fig. 2. the number of boxes per square unit is 1, therefore total number of boxes is 864 and number of visible boxes is 432. The number of boxes per square unit can be increased to any number that is square of a digit (1, 4, 9, 16, 25, 36, 49, 64 or 81). In Table I. Values of the metrics we have defined have been provided, for the cube densities 1, 4, 9 and 16, because those are the values which were utilized to evaluate frameworks. In Fig. 3. Provided are the images of ClayGL animation, with different cube densities.

TABLE I
APPLIED METRICS FOR THE SECOND ANIMATION

Cubes per Square unit	**Metrics**		
	Number of cubes per side of a large cube	***Number of visible cubes (3 sides)***	***Number of all cubes (6 sides)***
1	144	432	864
4	576	1,728	3,456
9	1,296	3,888	7,776
16	2,304	6,912	13,824

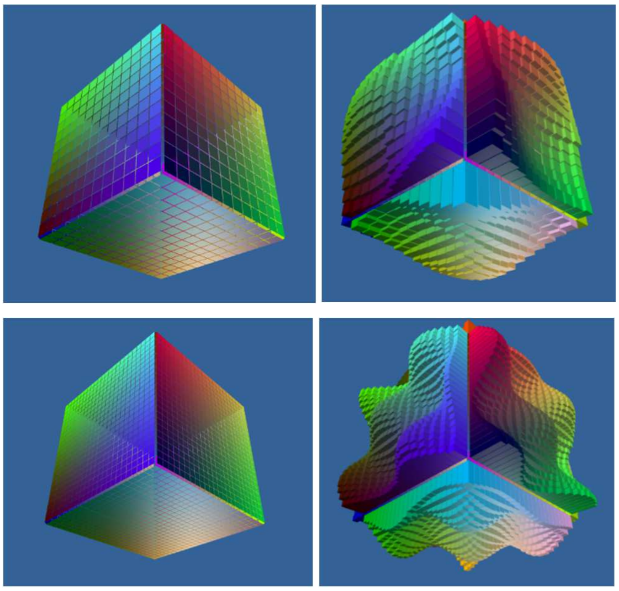


Fig. 3. ClayGL animation for different cube densities

- Wave animation

Wave animation can be either disabled (Fig. 3. top row) or enabled (Fig. 3. bottom row). Waves are formed by scaling the height of boxes at runtime. Values for the height of each box are calculated as a sine and cosine function of time in seconds. Wave animation can be enabled for any cube density.

- Position of the camera

The goal of this animation was to subject the frameworks to resource-demanding conditions, “stress test”, and evaluate the impact of those condition on graphics (visual quality of scene) and functionality aspect of animation (delay in the response to user initiated events).

A. Babylon.js

Babylon.js provides high visual quality despite utilizing basic object material for simple animation. It can be observed from Fig. 4. That cube object edges are sharp and straight, while sphere surface is smooth. Reflective surfaces of cube and sphere provide the visual impression of objects that have been described/enhanced with more than basic/default parameters, when in fact objects were created with default parameters. Upon closer inspection user can notice that there is small level of sphere distortion, but it does not diminish the overall impression by a great margin.

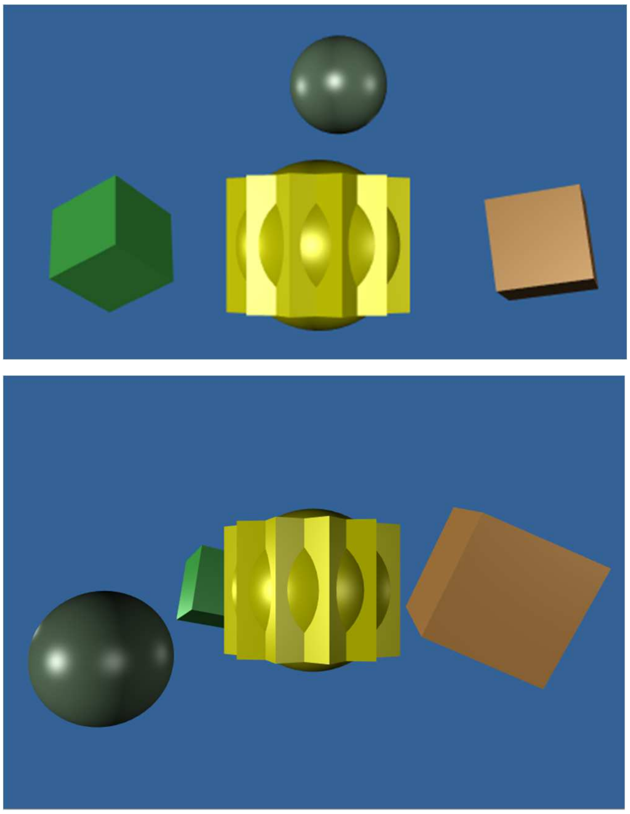


Fig. 4. First animation in Babylon.js

Babylon.js retains high quality graphics in more complex animations. Fig. 5. represents the second animation. Image on the Fig. 6. shows that, even with high density of cube (16 cubes per square unit) it can be observed that quality of the representation was not reduced, despite that from the perspective of the observer, visible cube sides are positioned at different angles and differently illuminated by directional lights. It can be noticed that quality of representation of boxes gradually lowers as boxes at greater distance from the camera are observed, but small individual boxes are clearly distinguishable on all visible cube sides. However, when different aspects of animation are manipulated at runtime, it becomes obvious that as the cube density increases, the animation displays considerable amount of delay in rendering new frames. The delay issues are apparent already at density of 9 cubes per square unit. Even though at densities higher than 16 cubes per square units Babylon.js animation is not realized (user changes are not realized) at reasonable time (more than 30 seconds) each frame that is realized retains the quality.

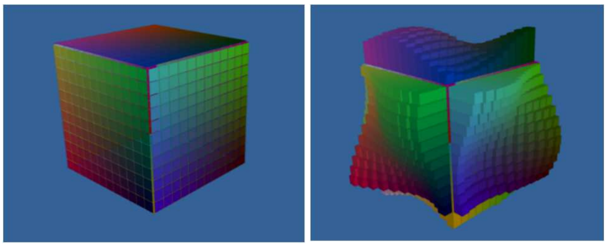


Fig. 5. Babylon.js visual representation of the second animation

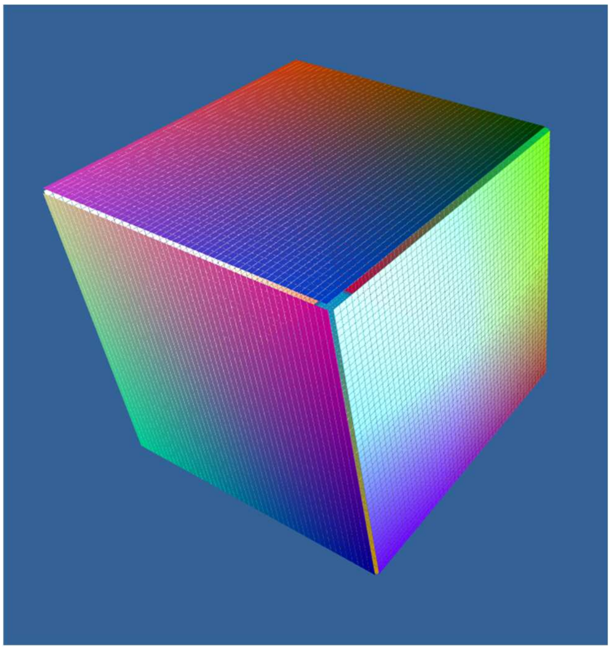


Fig. 6. Babylon.js keeps the high quality of the scene

B. *ClayGL*

ClayGL allows for the developing of high-resolution animations including just fundamental shapes and default parameters. The graphic quality is comparable to Babylon.js animation. Reflection of directional lights off the sphere surface is somewhat different from reflection off the sphere created in Babylon.js: In ClayGL sphere surface reflection of the light appears less intense and is of square shape, which gives off more realistic real-world impression. However, reflection of the central object appears too intense compared to surrounding objects, object color is overly exaggerated, and in that sense central object in Babylon.js has better graphics quality. As in Babylon.js, small sphere distortion is present, which is barely noticeable (Fig. 7. Bottom image).

ClayGL animation exhibits high quality graphics, on par with Babylon.js. Individual boxes are distinguishable for most of the visible cube area. The quality of graphic details is lower at visible cube areas farther from the camera, but that is inevitable given the number of elements. The aspect of animation where ClayGL is undoubtedly superior than Babylon.js is responsiveness to user-initiated animation actions in the high levels of animation complexity. When cube density is 9 or more, the rendering is not instantaneous, but is considerably faster than with Babylon.js animation, while retaining at least the same rendered frame quality (Fig. 3. And Fig. 8).

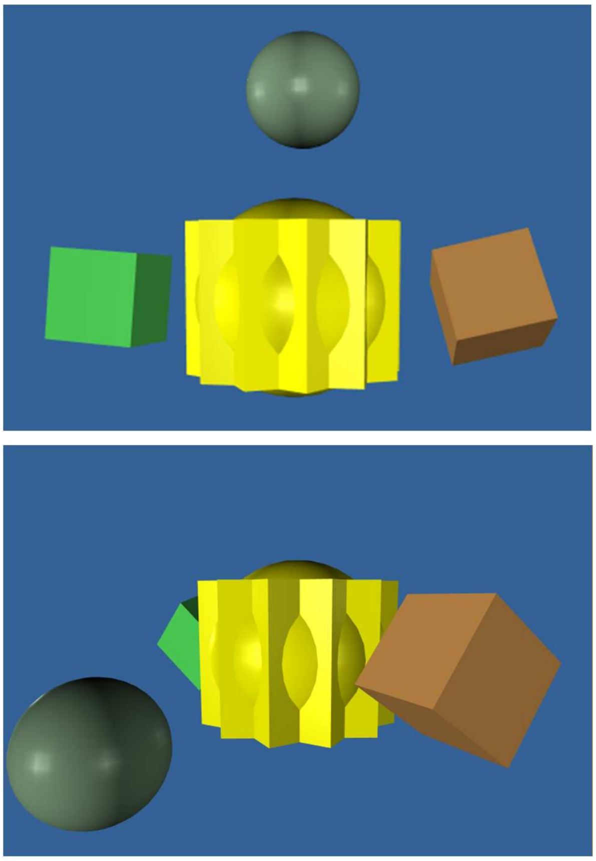


Fig. 7. First animation in ClayGL

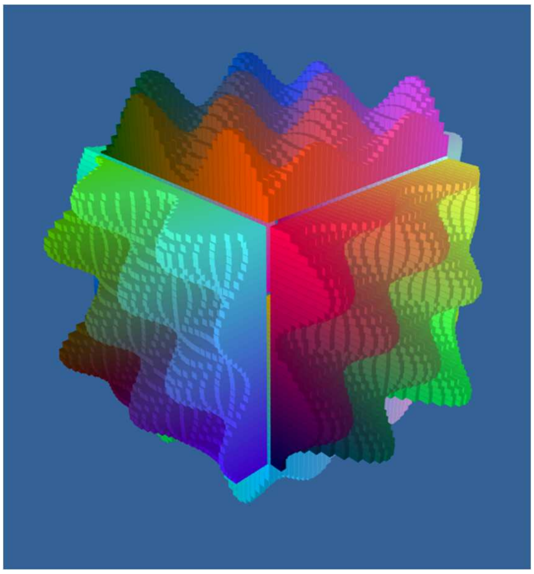


Fig. 8. Second animation in ClayGL

C. *Three.js*

Three.js animation (Fig. 9.) is of the poorest quality among the three frameworks. The distortion of the shapes near the edges of the display (Fig. 9. top) is substantial and can be noticed immediately, which could lower the consumers impression of the animation. The other evident issue is numerous "zigzag breaks" in the edges of cubes. Breaks appear to be present on object edges which are not aligned perfectly horizontally or vertically. Light reflection off the surfaces of sphere and cubes is good, but not on as good as in Babylon.js and ClayGL animations.

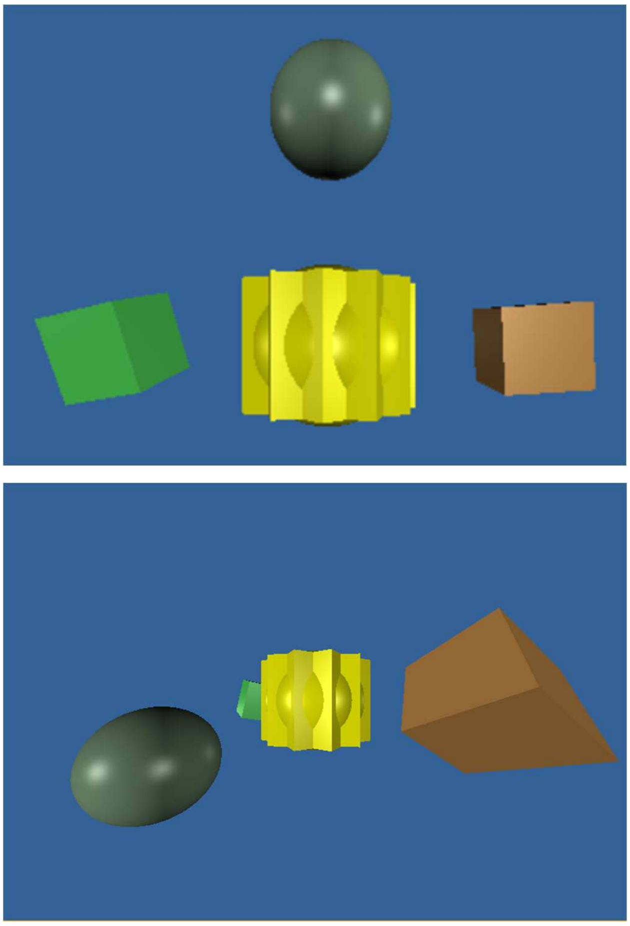


Fig. 9. First animation in Three.js

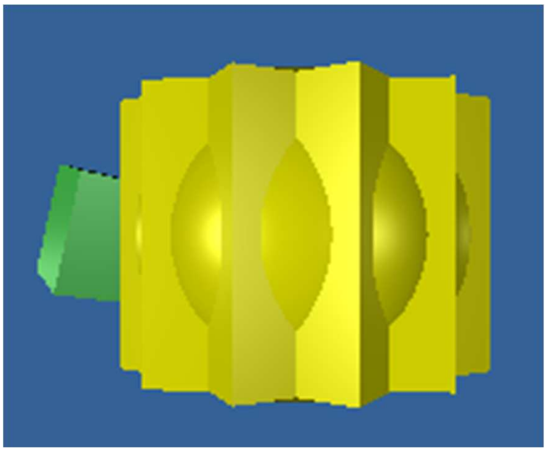


Fig. 10. Zigzag rendering of the edges in Three.js

Second Three.js animation displays the issues in the same aspects as with basic animation (Fig. 11.). Cube sides do not appear to orthogonal with respect to each other – cube object appears to have distorted shape. Individual boxes are fully distinguishable only on the small areas of visible cube sides that are closest to camera. As visible cube areas at greater distance from camera are inspected, it can be observed that they lose the overall visual/graphical/representation uniformity, that should be present in the scenes consisting of great number of identical objects arranged in order. Spaces separating the rows and columns have breaks and interruptions, and the "grid-like" pattern on visible cube sides (that is apparent in Babylon.js and ClayGL animations) is nonexistent. Once the wave animation is enabled, considerable amount of the detailed-oriented graphic representation shortcomings is corrected, possible due to dynamic nature of animation. Wave patterns are clear, individual boxes that were indistinguishable when the wave animation was disabled can now be clearly observed.

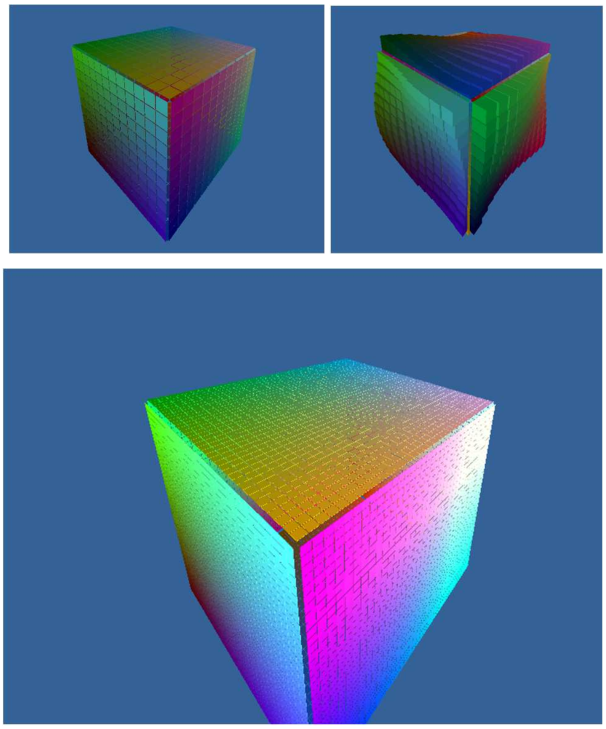


Fig. 11. Second animation in Three.js

IV. Comparative Analysis of The Observed Frameworks

In order to quantitatively present our observation, we have devised two tests to perform on animations. In both tests, the variable measured is "time needed to realize some user-initiated animation action". In other words, at the moment of animation change event was initiated by "keydown keyboard action" – the time measurement started, when the desired action was realized on the screen, time measurement ended. During the testing, animations were running in Google Chrome, time was measured by stopwatch and average of three measurements was recorded.

A. *First test*

In test 1, animation is subjected to cube density change, from the base case (1 cube per square unit) to 16 cubes per square unit. The time measurement started from the point of user-initiated change to cube density of 4 cubes per square unit. The moment the change is realized on screen, the second animation change is user initiated, to cube density of 9 cubes per square unit. Action is repeated for the third and last user-initiated change to cube density of 16 cubes per square unit, the moment the change is visible, the time measurement is completed. For every framework two scenarios are considered separately –wave animation disabled, and wave animation enabled. Measurements are recorded in Table II.

TABLE II
TIME MEASUREMENTS FOR THE FIRST TEST

Framework	Wave Animation disabled	Wave Animation enabled
ClayGL	2 s	2.1 s
Three.js	4.4 s	4.8 s
Babylon.js	6.4 s	32.2 s

Babylon.js is noticeably the slowest to render the scene. However, once the scene is rendered and animation is resumed, the uninterrupted (by user inputs) animation is not “lagging” any more than ClayGL and Three.js.

B. *Second test*

In order to realize the second test, new user-initiated event was defined. On user input, camera starts moving along one axis (position of the camera on two other axes remains constant). The size of camera step is 0.1 and one step is taken per realized animation frame.

TABLE III
TIME MEASUREMENTS FOR THE SECOND TEST, FOR 1 AND 4 CUBES PER SQUARE UNIT

Density	1	1	4	4
Wave animation	OFF	ON	OFF	ON
ClayGL	1.9 s	1.9 s	5.7 s	5.8 s
Three.js	2 s	2 s	4.7 s	5.2 s
Babylon.js	2 s	2.1 s	5.8 s	7.5 s

TABLE IV
TIME MEASUREMENTS FOR THE SECOND TEST, FOR 9 AND 16 CUBES PER SQUARE UNIT

Density	9	9	16	16
Wave animation	OFF	ON	OFF	ON
ClayGL	12.1 s	12.4 s	21.6 s	22.4 s
Three.js	10.4 s	11.4 s	17.9 s	19.8 s
Babylon.js	13.5 s	16.2 s	23.2 s	28 s

The movement ends when 100 steps are made (100*0.1=10 unit distance), or 100 consecutive frames are redrawn, in other words. The time recorded is from the moment the camera movement is initiated to the moment the camera is stationary. In order for the test to reflect the impact of rendering of complex animation scene with a lot of elements, most of the visible cube area is observable from user perspective, during the movement. Time measured is for the cube densities 1, 4, 9 and 16 boxes per square unit, and for each cube density two scenarios are recorded separately – wave disabled, and wave enabled. Measurements are recorded in Table III and IV.

Even though Babylon.js and ClayGL are slower that Three.js in the context of this test, Three.js still produces the observably lowest animation graphics quality: visible shape distortion, and poor detail quality. Side by side comparison for Babylon.js, ClayGL and Three.js, from left to right, 9 cubes per square unit, is shown in Fig. 12.

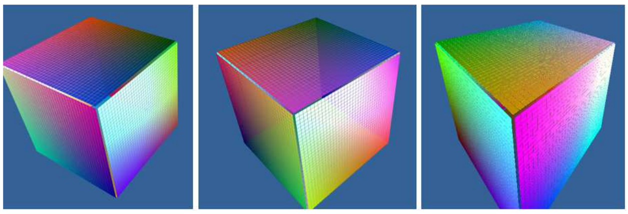


Fig. 12. Comparative side by side animation of the observed frameworks for the second test, for 9 cubes per suare units.

V. CONCLUSION

In this research, we analyzed current most popular frameworks for web 3D animation, based on JavaScript technology. There are numerous frameworks available, and there are many factors which could influence the selection of the appropriate framework for the specific needs.

All frameworks included in the analysis have the similar capabilities. However, each of the observed frameworks have both good and bad sides. For example, Three.js is the fastest of all three frameworks, but has the lowest animation quality. Three.js is also available on the market for a very long time, and has a large community of users and numerous examples and tutorials. ClayGL is one of the latest frameworks, lightweight and has excellent animation quality. On the other hand, ClayGL was slower in our tests, and generally has the least amount of the available documentation.

REFERENCES

[1] P. Pop, “Comparing Web Applications with Desktop Applications: An Empirical Study,” 2002.

[2] J.J. Garrett, “Ajax: A new approach to web applications,” 2005.

[3] M. Nordin, “An optimal solution for implementing a specific 3D web application”, 2016.

[4] E. Angel, D. Shreiner, “Interactive Computer Graphics with WebGL,” Addison-Wesley Professional, 2014.

[5] J. Moreau-Mathis, “Babylon. js Essentials,” Packt Publishing Ltd, 2016.

[6] Babylon.js homepage [online]: https://www.babylonjs.com/

[7] D. Catuhe, M. Rousseau, P. Lagarde, D. Rousset, “Babylon. js a 3D engine based on webgl and javascript”, 2014.

[8] ClayGL homepage [online]: http://claygl.xyz/

[9] Three.js homepage [online]: https://threejs.org/

[10] B. Danchilla, “Three. js framework. In Beginning WebGL for HTML5,” Apress, Berkeley, CA, pp. 173-203, 2012.

[11] Miew: 3D Molecular Viewer homepage [online]: https://lifescience.opensource.epam.com/miew/

[12] Publications in DiVA [online]: http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1228221&dswid=2287

SerbOCR - Optical Character Recognizer of the Serbian Cyrillic Alphabet

Jovan Vještica, Teodora Đorđević, Suzana Stojković, *University of Niš, Faculty of Electronic Engineering*

Abstract—**This paper presents a technique for optical character recognition of machine-written Serbian Cyrillic text based on deep Convolutional Neural Networks. The described OCR system makes use of distinct preprocessing, character segmentation and character classification modules. For preprocessing, an adaptive gaussian threshold was used, while character segmentation used horizontal and vertical projection as well as connected component analysis. The classifier itself is a deep convolutional neural network. The entire system gave good results, with scanned documents yielding strings with a Levenshtein distance of less than 1% of the string length to the actual value present on the image.**

Index Terms—**Optical Character Recognition; Neural Networks; Artificial Intelligence; Computer Vision; Image Processing**

I. Introduction

The oldest and most widespread mediums humans have used historically for storing information are visual mediums. As evidenced by a multitude of prehistoric cave paintings and ancient texts, humans have expressed their ideas, histories, and experiences in visual form since the dawn of humanity itself. Computer vision attempts to replicate the success of the human brain in recognizing optical patterns in order to decipher their meaning.

Optical character recognition, which in the modern context can be considered a subfield of computer vision, traces its roots to the first half of the 20th century, when the public was first introduced to practical machines capable of using optical signals as inputs. These machines, such as the Optophone [1], designed by Irish inventor Edmund Fournier d'Albe, were primarily created as aides for the visually impaired, thus allowing blind people to read regular print by producing a white noise of differing pitch.

In the post-war world of the second half of the 20th century, the market experienced a huge surge in both the availability and demand of consumer digital electronics. With the growing sophistication and power of these devices, as evidenced by Moore’s law, the need to properly digitize “legacy” stores of information, such as books, newspapers, and letters, arose. In order to improve the searchability and usability of these document stores, OCR techniques have been developed and applied in order to translate matrices of pixels into arrays of characters.

This paper presents a technique developed to analyze scanned documents written in the Serbian variant of the Cyrillic script. The approach that was developed contains following phases:

- image binarization by using an adaptive gaussian threshold,
- image rotation so that the text is parallel to the lower edge of the image,
- extracting the regions containing individual character based on connected component analysis and
- classification of image contents with a deep convolutional neural network.

II. Related work

Optical character recognition is a branch of computer vision that concerns itself with developing approaches to transforming a picture into characters. Typically, this process is broken up into multiple phases, each feeding its output into the input of the next [2]. At its most basic form, OCR involves two steps – generating rectangular regions which contain text and character extraction from those regions. This processing pipeline is often augmented in real applications by adding various preprocessing steps, mostly in the form of various filters. Postprocessing is also typically employed, primarily to increase the accuracy of the system for complex languages.

A. Binarization

Image binarization is one of the most important steps to consider when creating an OCR model. Image binarization attempts to map a matrix of pixels into a matrix of ones and zeros, with the intention of accentuating different structural components of individual letters by segmenting them from neighboring characters. The simplest approach of binarization is applying a fixed global threshold, but this approach typically gives poor results. Otsu’s method [3] is vastly more popular and is widely used in OCR applications before and during text segmentation. Otsu’s method determines a global threshold value that minimizes the foreground-background-class variances, which ensures good results when the image histogram has two distinct peaks. A new, iterative method based on Otsu’s method has been proposed, which classifies pixels into three classes – foreground, background and to-be-determined, where the latter class is iteratively distributed into the foreground and background, respectively [3].

Jovan Vještica is a student of Masters Studies at the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: jovan.vjestica@elfak.rs).

Teodora Đorđević is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: teodora.djordjevic@elfak.ni.ac.rs).

Suzana Stojković is with the Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia (e-mail: suzana.stojkovic@elfak.ni.ac.rs).

By applying an adaptive approach, not only was the pixel that is being processed taken into account, but also its immediate surroundings. The value being processed is either the mean value of the surrounding pixels or a gaussian-weighted sum [4].

A neural network-based approach has been proposed in [5] which uses the form of the histogram as well as a priori knowledge about the image as inputs to the general regression neural network that produces the final output.

B. Character segmentation

Character segmentation yields the locations of different characters presented on the image. Depending on the type of image being analyzed, different approaches may need to be applied. Identifying segmenting lines is important in document analysis, since it allows more efficient text segmentation. The most often used approach for line segmentation is horizontal projection, where different valleys indicate line breaks, typically represented as new line characters in documents. By analogy, an individual line segment can be split into word segments by applying vertical segmentation. The simplest and most direct approach to character segmentation is connected component analysis which can give satisfactory results when applied to preprocessed scanned documents [6]. Holistic methods exist which attempt to recognize words in their entirety thus evading the need for per-character segmentations [7].

C. Character classification

After properly segmenting the image, the final step is to apply the classification algorithm. Fundamentally, the goal of this step is to assign a class label based on an input matrix of fixed dimensions, where each class corresponds to each of the identifiable characters. Several methods can be applied here, including methods based on matrix-matching, feature engineering and neural networks [2]. A method based on neural network committees, which employ 7 different neural networks, has been applied very successfully to the MNIST dataset [8]. Additionally, there is no need to explicitly assign labels to characters – instead, a clustering algorithm may be applied to identify the different clusters, where each cluster represents a character. After postprocessing and human verification, the user can assign an explicit label in the form of an ASCII character to the cluster [9].

III. Image preprocessing and character segmentation

Since this paper deals with machine written documents that were scanned by a scanner, two key assumptions could be made – namely that all of the text lines present in the image were typed out at the same angle (this means that all the lines in the image were parallel) and that the lighting was consistent. The scanner itself, as well as the printing process of the document, can add noise to the scanned document. To denoise the image, preprocessing steps of edge extraction and thresholding are applied. Edge extraction is applied to remove the effects of background color and to better segment the individual characters [10]. The standard Canny edge detector from the OpenCV library is used, as it gave satisfactory results. The resulting edges are cleaned up and binarized by applying a global thresholding operation with a gaussian valued window.

After denoising the image, skew correction is applied. Skew correction is, in essence, a rotation of the image such that the text present on the rotated image spans horizontally instead of at an angle [11]. Since the image has been properly denoised and binarized, this is achieved by computing the minimal rotated rectangle that contains colored pixels. Next, an approach based on horizontal and vertical line projections is adopted to extract the different regions of interest that contain potential characters, while also applying connected component analysis to ensure that no characters are split.

To extract the regions containing characters, a horizontal projection to identify the text lines is applied. The begging of each line is denoted as a sharp decrease in the median value of the pixels along the scan line, while the end of a line is denoted as a sharp increase in the value. Similarly, the same method of splitting vertically in order to isolate individual characters can be applied. Since machine-written Serbian does not contain any ligatures nor does it contain any pair of letters that are connected, this approach works well. A potential problem that can arise in this phase of processing is the splitting of individual characters. To combat this, a variant of connected component analysis is applied by disallowing a vertical line to split a region if there are any colored pixels that have neighboring colored pixels in the next column. Additionally, once more a horizontal projection is applied on the newly extracted window. This is because lines will typically contain both capital and regular letters, which have varying heights. By choosing a fixed height for the entire line being analyzed, all regular letters would end up having a large blank upper portion, which would convey no information to the classifier later. Moreover, even regular letters in Serbian Cyrillic come in vastly different heights (for instance a and ђ), which becomes very evident in certain fonts.

As the last step in character extraction, the defined region is resized to a fixed 32 x 32 resolution. This is mandatory since the classifier can only classify 32 x 32 tensors. The resizing algorithm applied is based on bilinear interpolation, which is one of the commonly used algorithms for image resizing [12].

IV. Character classification in SerbOCR

After extracting all the regions that contain characters, classification is applied to transform the image, which is in this stage a 32 x 32 binary matrix, into a character. Several different classifier architectures were considered. Firstly, a standard multi layer perceptron network was used for classification, but it had severe drawbacks – namely, the loss function was never optimized fully, or the network would severely overfit, causing low precision on the testing data. After this, a shallow but wide convolutional network was considered, which gave acceptable results. Ultimately, a deeper network gave the best results. This architecture has found wide applications in image classification [13].

Before training the classifier, a dataset needed to be constructed. By applying the document segmentation algorithm, as described earlier, it was possible to isolate the individual characters present on the images that comprised the dataset. This ensured that both the format and position of the individual character in the region would be similar to what the classifier would encounter in a production environment, as well as having the added benefit of preserving the noise present on the initial image. This region was then classified using Tesseract, an open source OCR solution [14]. Based on the results of this classification process, the image was then stored onto the disk in the directory corresponding with the given class. In total 82 distinct classes were used, corresponding to the digits 0-9, all of the letters of the Serbian alphabet in both their regular and capital variants, as well as several special characters, such as parentheses, the exclamation mark, the question mark, and others. To increase the robustness of the classifier, all of the images in the dataset had a salt-and-pepper noise applied to them. This additional noise was meant to force the classifier to actually extract meaningful features from the images being analyzed. Taking into account the statistical properties of natural languages, it was expected that certain classes would be heavily favored, while others would conversely be neglected. Some classes, such as the vowels, were overrepresented in the dataset, while others were almost entirely absent. To combat this, data augmentation was applied to the underrepresented classes, such as applying paddings, slight stretches or small rotations and randomly cropping the resulting image to bring it to the needed size [15]. Each augmented data point used its own salt and pepper noise. Since the overrepresented classes had certain data points that were virtually identical to each other, a random sampling of these classes was applied. This dataset was then split into a training, testing and validation dataset, with an 8:1:1 ratio. In total, approximately 50 000 data points were used to train the classifier.

The most natural way of encoding the labels for the dataset was using the categorical encoding strategy (one-hot encoding). This approach unraveled the label column into 82 binary columns, where a value of 1 at index *i* denoted that the data point was a member of class *i*. Naturally, the new label vector can only feature a single 1 value, with the rest of the values being zeroes.

The classifier itself is a feed forward deep convolutional neural network. The network consists of four blocks, with an additional global maximum pooling layer and a fully connected layer at the end. Each neural network block consists of a two-dimensional convolution layer, followed by a batch normalization block. The entire network has been implemented using the keras library for interacting with the tensorflow backend.

The convolutional layers act as feature extractors, which operate by applying a learned convolution kernel to the data being processed. A standard square kernel with a dimension of 3 was used. Each convolutional layer uses the rectified linear unit activation function. By applying a padding operation and setting the convolution stride to 2 in both directions, a down sampling operation was applied by reducing the dimensionality of data by a factor of two in each dimension. Thus, there is no need to apply a maximum pooling operation, although this is common.

Batch normalization is a technique that can be used in convolutional neural networks to increase their training speed and stability and is often used to regularize deep convolutional models as a powerful alternative to the classic dropout approach [16]. A batch normalization layer is applied after each convolutional layer.

The penultimate layer in the neural network is a global maximum pooling layer, which reduces the dimensionality of the network to a single extracted feature vector. This vector serves as the input to the final layer of the neural network, which is a fully connected layer. This layer has 82 neurons, each corresponding to a potential class. The activation function of the final layer is the softmax activation function. This ensures that the 82 output values of the network can be interpreted as probabilities, specifically as the probability that image being classified belongs to the class associated with the neuron. The final value of the classification is inferred as the most probable class.

For training purposes, a loss function needed to be defined. Since at this stage this is essentially a multiclass classification problem, a natural candidate for the loss function is the categorical crossentropy function [17]. As the final performance metric, the validation loss is chosen instead of training loss, choosing to save the model version which minimized the validation loss rather than the training loss.

V. Evaluation

The implemented OCR solution was evaluated by using a series of documents issued by the Council of the University of Niš. These documents contained text with various formatting options, such as bold letters, italic letters, letters of different font size and letters of different fonts in general. In addition, all the documents are written in the Cyrillic script.

A. *Character segmentation*

Character segmentation was the part of the pipeline that introduced most errors. Typically, the errors were introduced by character pairs, that when printed together had no space between them and appeared as a single character. The most represented pair that fits this description was ‘**ст**’. Also, the errors were common with pairs: ‘**гл**’, ‘**УЈ**’, ‘**Уј**’, ‘**ту**’, ‘**ут**’, ‘**уд**’, etc.

TABLE I
Character Segmentation

Paragraph	Length	Estimated length	Success of segmentation
Paragraph1	228	202	88%
Paragraph2	169	154	91%
Paragraph3	351	322	92%

In Table I is shown the success of segmentation that was achieved. On average the segmentation algorithm detected less characters that were present. This was because of implemented connected component analysis, which ensured that a single character is never detected as many characters. The average accuracy of the segmentation algorithm was 90%.

B. *Classifier accuracy*

The classifier achieved good accuracy on the train and validation sets. The accuracy was over 96% on the validation set. The accuracy on the test set was lower owing to the fact that not all characters were equally present in the training set. For instance, the letter ‘**а**’ was over a hundred times more present than the letter ‘**џ**’. The classifier mainly produced errors when it came to underrepresented letters and symbols in the training set such as ‘**џ**’, ‘**Ћ**’, ‘**Ж**’, ‘**ж**’, ’**;**’. Since these symbols are also statistically underrepresented in the Serbian language the overall classifier accuracy was not lowered substantially achieving a final accuracy of 95%.

C. Paragraph level analysis

The measure that was used to evaluate the performance of the system is the Levenshtein distance. The Levenshtein distance is a metric that measures how many edits need to be applied to a certain string in order to transform it into a different string [18]. A valid edit is considered an insertion, deletion, or substitution.

The following table shows the Levenshtein distance in relation to the string length on a per-paragraph basis for a selected document. Seeing as the Levenshtein distance depends primarily on distance, the following table displays both the paragraph length, expressed as the number of characters present in the paragraph, and the Levenshtein distance between the detected content of the paragraph and the actual content.

TABLE II
PER-PARAGRAPH LEVENSHTEIN DISTANCES

Length	CNN	Deep CNN
308	42	19
660	69	25
390	55	26
749	69	28

The first paragraph was very text dense, with not a small number of whitespace characters and special symbols. The second paragraph was structurally similar to the first, containing mostly plain text. Paragraph 3 contained a high number of punctuation characters. The fourth paragraph was like the first two.

The deep CNN consistently outperformed the shallow CNN, having on average fewer mistakes than the shallow CNN in every paragraph used for analysis. This was especially noticeable in examples 2 and 4, where the deep CNN outperformed the shallow CNN by a factor of 2.5.

As can be seen, most of the errors were induced by special symbols and whitespace characters. Special symbols were severely underrepresented in the training dataset, and after resizing, the classifier struggled to identify them correctly. White space characters get detected as part of the text segmentation process, and not the classifier, and thus improvement in the dataset would not improve this.

VI. CONCLUSION

This paper has outlined an OCR system created for the Serbian Cyrillic script. First, it is described how to perform the character segmentation from a scanned, machine written document. After this, the way the dataset used to train the classifier is generated, is described. Finally, the architecture of the classifier is shown and explained.

From all the above, it is shown that general purpose OCR techniques can be effectively specialized to be applied to machine written Cyrillic. There is still a lot of opportunity to improve the system described in this paper, primarily in the way special characters such as white spaces and others are handled. The analysis of cursive, handwritten Cyrillic also poses several new and interesting problems.

ACKNOWLEDGMENT

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

REFERENCES

[1] Albe E. E. Fournier, L. O. Joseph, “On a type-reading Optophone”, Proceseedings of the Royal Society, vol. 90, no. 619, pp. 373-375, July 1914

[2] S. Singh, “Optical Character Recognition Techniques: A Survey”, *Journal of Emerging Trends in Computing and Information Sciences*, vol. 4, no. 6, pp. 2009-2015, June 2013

[3] N. Senthilkumaran, S. Vaithegi, “Image segmentation by using thresholding techniques for medical images”, *Computer Science & Engineering: An International Journal*, vol.6, no.1, February 2016

[4] S. Khurana, “Comparative Study on Threshold Techniques for Image Analysis”, *International Journal of Engineering Research & Technology*, vol. 4, no. 6, June 2015

[5] J. Lázaro, J. L. Martin, J. Arias, A. Astarloa, C. Cuadrado, “Neuro semantic thresholding using OCR software for high precision OCR applications”, *Image and Vision Computing*, vol. 28, no. 4, pp. 571–578, 2010

[6] A. Kaur, S. Baghila, S. Kumar, “Study of various character segmentation techniques for handwritten off-line cursive words: a review”, *International Journal of Advances in Science Engineering and Technology*, vol. 3, no. 3, pp. 154-158, July 2015

[7] R. G. Casey, E. Lecolinet, “A survey of methods and strategies in character segmentation”, *IEEE Transactions on Pattern Analysis and Machine Inteligence*, vol. 18, no. 7, pp. 690-706, August 1996

[8] D. C. Cires, U. Meier, L. M. Gambardella, J. Schmidhuber, “Convolutional Neural Network Committees For Handwritten Character Classification”, 2011 International Conference on Document Analysis and Recognition, Bejing, China, pp. 1135-1139, September 2011

[9] G. Vamvakas, B. Gatos, N. Stamatopoulos, S. J. Perantonis, “A Complete Optical Character Recognition Methodology for Historical Documents”, Document Analysis Systems, IAPR International Workshop, pp. 525-532, September 2008

[10] M. Zaharescu, I. C. Petrescu, “Edge detection in document analysis”, *Journal of Information Systems & Operations Management*, vol. 7, pp. 156-165, May 2013

[11] R. Ahmad, S. F. Rashid, M. Z. Afzal, M. Liwicki, A. Dengel, T. Breuel, “A novel skew detection and correction approach for scanned documents”, Document Analysis Systems, IAPR International Workshop, April 2016

[12] E. E. Danahy, S. S. Agaianb, K. A. Panettaa, “Algorithms for the resizing of binary and grayscale images using a logical transform”, *Image Processing: Algorithms and Systems V*, vol. 6497, February 2007

[13] W. Rawat, Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review”, *Neural Computation*, vol. 29, pp. 1-98, June 2017

[14] R. Smith, “An Overview of the Tesseract OCR Engine”, Ninth International Conference on Document Analysis and Recognition, vol. 2, pp. 629-633, September 2007

[15] A. Starynska, R. L. Easton, Jr. D. Messinger, “Methods of data augmentation for palimpsest character recognition with Deep Neural Network”, 4th International Workshop on Historical Document Imaging and Processing, pp. 54-58, November 2017

[16] E. Chai, M. Pilanci, B. Murmann, “Separating the Effects of Batch Normalization on CNN Training Speed and Stability Using Classical Adaptive Filter Theory”, February 2020

[17] Z. Zhang, M. R. Sabuncu, “Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels”, May 2018

[18] G. Cormode, S. Muthukrishnan, “The String Edit Distance Matching Problem with Moves”, *ACM Transactions on Algorithms*, vol. 3, no. 1, February 2007

OpenGL Specular Reflections Caused by Light Source Placed Below Shadowed Object

Dženan F. Avdić, Dejan D. Rančić, Aldina R. Avdić and Petar Ć. Spalević

Abstract— **Reflections and shadows are crucial for enhancing scene realism and providing important visual cues. In recent years, many important contributions have been made in modeling both for shadows and for reflections which are consequences of the presence of objects made of special material, with the mirror effect. Shadows are created by a light source that is placed above the shadowed object. Those shadows are placed on the surface where the shadowed object is placed. But, if a point light source is placed below a shadowed object made of a material that reflects light, the phenomenon that is created on that surface is not a shadow. This specular reflection is a topic that will be described in this paper, on the example of circle and sphere object.**

Index Terms— **Circle, OpenGL, Point light source, Reflections, Shadows, Sphere.**

I. Introduction

Making a virtual reality using computer graphics is a fundamental part of implementing computer games, CAD software, software dedicated to various types of simulations, etc. The existence of phenomena that are caused by the presence of light sources in the scene, such as shadows and reflections, is essential for the improvement of visual perception. Since their using depicts relations between objects on the scene, users can sense more exactly the distance between two virtual objects. So, those components enhance the 3D impression to users can get a better immersive 3D feeling [1, 2].

Shadow is an area where direct light from a light source cannot reach due to obstruction by an object. It occupies all of the space behind an opaque object with light in front of it. The cross-section of a shadow is a two-dimensional silhouette or reverse projection of the object blocking the light.

For a non-point source of light, the shadow is divided into the umbra and penumbra. The umbra is a dark component of shadow, and its creation is a consequence of the fact that there is the area of a shadowed object that is not visible from any part of the light source. Light sources with their area can make another component of shadows which is less dark from the umbra. It happens when the area of a shadowed object that can receive some but not all, of the light from the light source [3].

So, a point source light has no penumbra, since no part of a shadowed object can receive only part of the light [4].

Reflection is the change in direction of a light ray at an interface between two different media so that the light ray returns into the medium from which it originated [5]. The law of reflection says that for reflection the angle at which the ray is incident on the surface equals the angle at which it is reflected.

If the reflecting surface is very smooth, the reflection of light that occurs is called specular or regular reflection. Otherwise, if the reflecting surface is rough, reflection is diffuse. The laws of reflection are as follows (shown in Fig. 1):

- The incident ray, the reflected ray, and the normal to the reflection surface at the point of the incidence lie in the same plane.
- The angle which the incident ray makes with the normal is equal to the angle which the reflected ray makes to the same normal.
- The reflected ray and the incident ray are on the opposite sides of the normal.

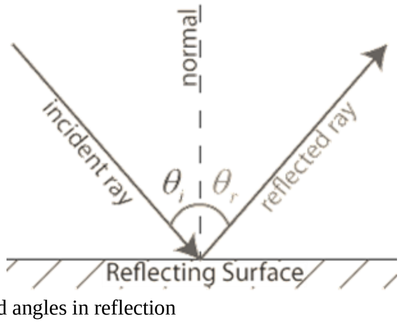


Fig. 1. Rays and angles in reflection

Refraction is the change in direction of a wave due to a change in its medium. It is essentially a surface phenomenon. Due to the change of medium, the phase velocity of the wave is changed but its frequency remains constant. This is most observed when a wave passes from one medium to another at any angle other than 90° or 0°. Refraction of light is the most observed phenomenon, but any type of wave can refract when

Dženan F. Avdić is with State University of Novi Pazar, Department of Technical Sciences, Vuka Karadžića bb, 36300 Novi Pazar, Serbia, (E-mail: dzavdic@np.ac.rs)

Dejan D. Rančić is with University of Niš, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, (E-mail: dejan.rancic@elfak.ni.ac.rs)

Aldina R. Avdić is with State University of Novi Pazar, Department of Technical Sciences, Vuka Karadžića bb, 36300 Novi Pazar, Serbia, (E-mail: apljaskovic@np.ac.rs)

Petar Ć. Spalević is with University of Pristina, Technical Sciences, Knjaza Milosa 7, Kosovska Mitrovica, Serbia, (E-mail: petar.spalevic@ftn.pr.ac.rs)

it interacts with a medium, for example when sound waves pass from one medium into another or when water waves move into the water of a different depth.

Refraction is described by Snell's law, which states that for a given pair of media and a wave with a single frequency, the ratio of the sinus of the angle of incidence θ1 and angle of refraction θr is equivalent to the ratio of phase velocities (v1 / v2) in the two media, or equivalently, to the opposite ratio of the indices of refraction (n2 / n1) is given with eq.1:

$$\frac{\sin\theta_1}{\sin\theta_r} = \frac{v_1}{v_2} = \frac{n_2}{n_1} \tag{1}$$

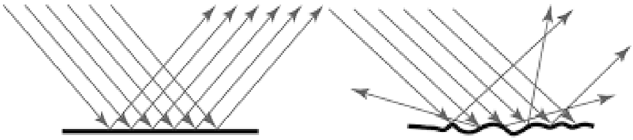


Fig. 2. Specular and diffuse reflection

Reflection in computer graphics is used to emulate reflective objects like mirrors and shiny surfaces.

OpenGL does not support reflections directly, but some kinds of these can be implemented in several ways. The methods vary in their difficulty to implement, their performance, and the quality of their results. These qualities depend on two parameters: the complexity of the observed object and the complexity of the scene where this object is placed [6, 7, 8].

The approach described in this paper supports makes the formation of specular reflection easier, considering the events that occur when objects are exposed to the light. In this example, the light source is a point source of light, and the surface, on which the light rays to form a shadow, is flat.

The aim is to determine the minimum number of characteristic rays coming from the light source and points out that these rays pass, which are essential for the formation of specular reflection on the surface [9]. Reflections are created for two types of objects (circle and sphere).

The paper is organized as follows. The first section provides an overview of the existing methods, with emphasis on their advantages and disadvantages. The next section presents the mathematical base for the formula derivation, which allowed finding characteristic points for drawing reflections. Then follows the description and view the experiment in which an application is created based on the formula, and the results are displayed. Finally, conclusions and trends in future work are given.

II. RELATED WORK

Reflection is accomplished in a ray trace renderer by following a ray from the eye to the mirror and then calculating where it bounces from, and continuing the process until no surface is found, or a non-reflective surface is found. Reflection on a shiny surface like wood or tile can add to the photorealistic effects of a 3D rendering.

Polished - A polished reflection is an undisturbed reflection, like a mirror or chrome.

Blurry - A blurry reflection means that tiny random bumps on the surface of the material cause the reflection to be blurry [10, 11].

Metallic - A reflection is metallic if the highlights and reflections retain the color of the reflective object.

Glossy - This term can be misused. Sometimes it is a setting which is the opposite of Blurry. When "Glossiness" has a low value, the reflection is blurry [12, 13].

Reflections have applications in the study technique. This is described in [14] but in the case of the ray-tracing technique.

III. IMPLEMENTATION OF SPECULAR REFLECTIONS

The goal of this research is to establish a single library with methods for adding specular reflections after drawing an object, in case that the light source is below that object. As the reflections will be realistically drawn for each of the primitives, complex objects consisting of 2D and 3D primitives will have their own reflection which will be formed by the reflection of its component objects. This approach uses the phenomena that occur in the real-world in the case when the scene is lighted by a point source of light, which can move along the axis of the coordinate system.

The effectiveness of these ways of the rendering of reflections is in the fact that there is a need to calculate minimum points relevant for drawing them. The second circumstance is the use of the fact that there are similarities between reflections of the primitives. A sphere is made by the rotation of a circle around one of the axes. So, with finding a method for forming the reflection for 2D objects, it is easy to create the reflection for the 3D object.

The paper describes the following case. For two objects (circle and sphere) that are located away from the surface for a distance d, the reflection is on the surface that is created by the point light source with the possibility of movement. The surface is flat. The light source can move the X and Y axes.

Due to many options and restrictions for showing 3D system on 2D paper, some less relevant parameters for calculating the are taken as fixed:

- The light source is always below the object (yi < yp + hp)
- X and Z coordinates of the object are point 0.
- xi, yi, zi are the coordinate position of the light source (zi = 0)
- r is the radius of the circle and the sphere.
- d is the distance between the object and the surface.
- xp, yp, zp are coordinates of the center point of the object (for circle it is the circle's center, also for the sphere) (xp, zp = 0).

A. A Circle

In this case, the object which is considered is a circle. The events that occur when the light source is below the circle which is set parallel to the flat surface are monitored. This means that the value of y coordinate object (circle) is higher than the value of y coordinate sources of light. A phenomenon which is then displayed on the surface is not a shadow of the object, but rather a reflection object (circle). It examines the case of the circle, as it is in the structure for other elements, so the reflection of more complex objects which have a circle in its basis calculates using the reflection of the circle.

According to the theoretical basis of which is given in the introductory part, the light is reflected from all points of a case on the same angle at which it came to that point. As stated in working with composite shadows [10], there is no need to look at all points and looking for reflections of each item that belongs to the circle. This would make rendering significantly complex. What we concluded lightening circular object of negligible thickness mounted parallel to the surface, if the light source is below the observed object, is that the shape of its reflection is also a circle.

Two cases are considered: when the light source is just below the circle (x = xp = 0) (Fig.3.a) and when the light source is placed to the left or right of the object (Fig.3.b).

An important point is the center of the reflection. Also, important information needed for rendering is the size of the radius of the reflection. Coordinates of the center of reflection could be found as follows. First, we need to find a point that is the symmetric center of the light source relative to the center of the circle, i.e. in this case to the y-axis. In this way, the ray is reflected light is passing right through the center of the circle and the symmetric point. The intersection of the reflected rays of light and surface reflectance is the center of the circle on the surface.

If the source of light just below the object, the coordinates of the center of reflection are known and they are coordinates of the beginning point (0, 0, 0). If not, the formula for calculating the center of the reflection is given with (2).

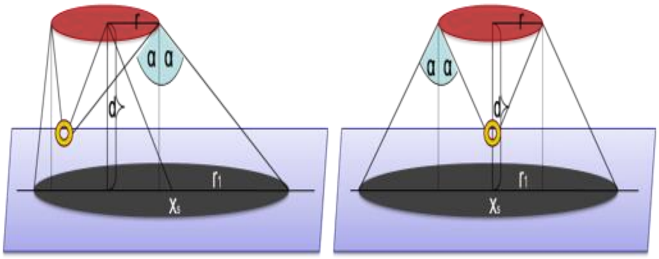


Fig. 3. A reflection of the circle on the plat surface using a point light source: left) Xi=Xp=0 right) Xi≠Xp

$$x_s = \frac{d \cdot x_i}{y_i - d} \tag{2}$$

To calculate the length of the radius of reflection, we should have to watch the ray of the light that is reflected from two points on the circle, i.e. to watch endpoints diameter circle when the x coordinate of a diameter equal to zero. The process of finding reflected rays is carried out in the same manner as in the case of finding reflected rays from the center point of the circle.

The equation for calculating the radius of the reflection is given in (3). From this formula, the length of the radius depends on r, yi, and d.

$$r_1 = \frac{-d \cdot r}{y_i - d} + r \tag{3}$$

The reflection of the circle is a circle with its center at the center is (xs, 0, 0), and which radius is r1.

Including the equation h=yi-d, equations 2 and 3 can be expressed as eq.4 and eq.5:

$$x_s = \frac{d \cdot x_i}{h} \tag{4}$$

$$r_1 = \frac{dr}{h} + r \tag{5}$$

B. A Sphere

A sphere is a 3D object that is created by circle rotation around the X-axis but shifted to r on Y ax.

When we look at creating a sphere reflection caused by placing light sources under the ball, what can be concluded is that a large part of the sphere is not visible from the light source. Light affects only one part of the sphere. The reflection of the sphere is equal to the reflection of this part, i.e., its largest circle. The problem of creating reflection is reduced to finding the intersecting circles, from whose center and endpoints light rays bounce from the center of the point source of light. The condition that must be fulfilled to have a reflection of the sphere which has its shape, i.e. not to waste all over the place, is that the absolute value of x coordinates of the light source is less than the radius of the sphere, in case of the sphere whose center is at (0, y), as shown in Equation 6.

$$|x_i| < r \tag{6}$$

In this case, it is necessary to find a tangent to the circle sphere whose center is to coordinate the beginning, and radius of the x and y-axis, while the z = 0. Because the ray that affects the circle from the point source of light is a tangent to the circle. We need to find a part of the sphere, which is enough for the formation of reflection, and he is determined with intersecting points from tangents from the point source of light and the sphere (i.e. the circle which we observe). As we point out that withdrawing tangent and we know the formula for a circle that is tangent, it is necessary to find the right equation of tangent (in this case, the two tangents).

The equations of tangents are obtained as a solution of the system of equation (7).

$$\begin{gathered} y = kx + l \\ x^2 + y^2 = r \end{gathered} \tag{7}$$

Tangent formula (8) is found if we consider the condition that some line is a tangent to the circle centered at the point O [15].

$$\begin{gathered} k_{1/2} = \pm \frac{\sqrt{-4(1-x_i^2)(1-(y_i-d)^2)}}{2(1-x_i^2)} \\ l_{1/2} = y_i - k_{1/2} x_i \end{gathered} \tag{8}$$

Characteristic points of the reflection are given with (9):

$$x_{1/2} = -\frac{\left(k_{\frac{1}{2}}(l_{1/2}-d)\right)}{\left(1-k_{1/2}^{2}\right)}$$

$$y_{1/2} = k_{1/2}x_{1/2} + l_{1/2} \tag{9}$$

Then the intersecting point of the tangent and the circle is found as a solution of the equation of circle and equations tangent of the circle because the intersecting point is at the same time and on the circle and the tangents, which means that it satisfies both formulas.

Therefore, the algorithm to be followed when creating a reflection of the sphere is as follows:

1) to find the equation of the tangent from the light source to the circle belonging to the sphere of the x, y plane, where z = 0;

2) to find common points that belong to tangent and circle.

3) to find a symmetric beam or reflected beam for the intersection points (in the same way as in the circle), as well as with the center of the intersecting circle.

There are two cases: when the light source is just below the sphere and when the light source is right/left are shown in Figure 4.

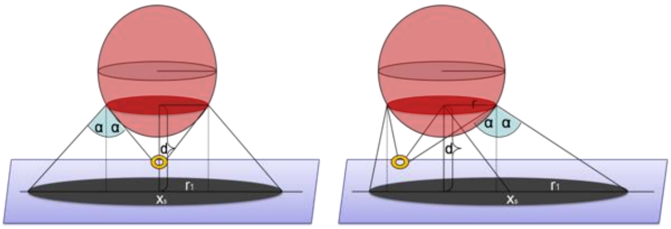


Fig. 4. A reflection of the sphere on the plat surface using the point light source: left) xi=xp=0 right) xi≠xp

IV. EXPERIMENTS

To confirm the validity of practical formulas and check the speed and the reality drawing shadows created a C++ application that uses OpenGL. The application considers the above limitations. The application has the following features:

Changing of a shadowed object, which could be a circle or sphere (keys K, L)

Changing the position of the camera, by increasing or decreasing x, y, and z coordinates (keys Y, X, C, A, S, D)

Changing the position of the point light source, by increasing and decreasing x and y coordinates (keys V, B, N, M).

The results for each element are given with the following screenshots (Fig. 5). The subject of future research will be troubleshooting the glare of the light source where y coordinate of the light source is less than y coordinate shadowed object, i.e. when the light source is below the shined object.

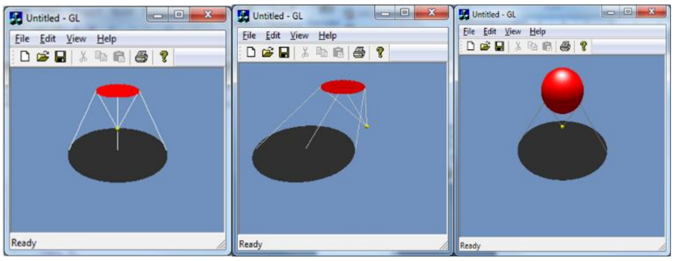


Fig. 5. The implementation of shadows in the OpenGL C++ application

V. CONCLUSIONS

By projecting all points on the surface slows the drawing of the reflections. Therefore, it is enough to find the minimum number of characteristic points, and the real reflection will not be disturbed, and its rendering will be easy and cheap.

In the future, the focus will be on the finding methods for making reflections remaining primitives that are commonly used. And other cases will be analyzed: the case when the object can move, what happens if there are more sources of light or more objects in the scene, or if the surface is not flat.

After analyzing all the conditions in which the object can be found, the output should be a unique and efficient technique for forming real OpenGL specular reflections.

REFERENCES

[1] M. Haller, S. Drab, W.Hartmann, "A realtime shadow approach for an Augmented Reality application using shadow volumes" in VRST '03 Proceedings of the ACM symposium on Virtual reality software and technology, (Osaka, Japan), pp. 56–65, October 2003.

[2] E. Kajan, S. Djordjevic-Kajan, N. Blagojevic, Z. Radenkovic. "Towards Computer Graphics for Strategic CIM", Automatika, vol. 31, no. 1-2, pp. 19-24, 1990.

[3] T. McReynolds, D. Blythe, Advanced Graphics Programming Using OpenGL, The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, ISBN: 1-55860-659-9, 2005.

[4] C. Wyman, C. Hansen. „Penumbra maps: Approximate soft shadows in real-time" in Proceedings of Eurographics Symposium on Rendering (Aire-la-Ville, Switzerland), pp. 202–207, June 2003.

[5] M. D. McCool, "Shadow Volume Reconstruction from Depth Maps," ACM Transactionson Graphics, vol. 19, no. 1, pp. 1-26, 2000.

[6] T. Akenine-Möller, E. Haines, N. Hoffman, Real-Time Rendering, A K Peters, 3rd edition, 2008.

[7] R. Fernando (ed.). GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics, Reading, MA: Addison-Wesley Professional, 2004.

[8] J. Kessenich, D. Baldwin, R. Rost, „The OpenGL Shading Language" (Version 1.051), 3DLabs, Inc., Egham, Surry, Feb. 2003, www.opengl.org/documentation/oglsl.html

[9] Pljaskovic, D. Avdic, P. Spalevic, D. Rancic, "Sphere and Cone Composite Realtime Shadows in OpenGL", Ictf, Sarajevo, May 2013.

[10] M. Ashikhmina, A. Ghoshb, "Simple Blurry Reflections with Environment Maps", Journal of Graphics Tools - Special on hardware-accelerated rendering technique, vol. 7, no. 4, pp. 3-8, 2002.

[11] S. R. Buss, 3D Computer Graphics, A Mathematical Introduction with OpenGL, Cambridge University Press, 2003.

[12] Wikipedians, 3d rendering, PediaPress.

[13] T. Davidovic, J. Krivanek, M. Hasan, P. Slusallek, K. Bala, "Combining global and local virtual lights for detailed glossy illumination", ACM Trans. Graph., vol. 29, no. 6, 2010.

[14] A. Pomi, P. Slusalle, "Interactive Ray Tracing for Virtual TV Studio Applications", Journal of Virtual Reality and Broadcasting, vol. 2, no. 1, 2005.

[15] P.K. Jain, A Textbook of Analitycal Geometry of Three Dimensions, Age New International, 2005.

Singular Value Decomposition (SVD) as an Approach for Digital Image Compression

Miloš Ilić, Petar Spalević, Dragana Stojnev, Nebojša Stojanović, Aleksandra Stojnev Ilić

***Abstract*—The problem that we tried to solve is the processing of digital images collected by photographing agricultural land from the air. Images created in this way are most often used in the process of assessing the condition of sown crops on productive agricultural land. In order to obtain high quality images, anemone aircraft equipped with high-resolution cameras are in use. High-resolution photos usually require a large storage space, so we must work on their compression, which must not affect the quality of the photos themselves. Singular Value Decomposition has taken a significant place in the implementation of image processing applications in recent years. It's importance is especially reflected in the fact that it provides quality compression of images, and object recognition on images. In this paper, we propose a survey for the SVD as an efficient transform method, which can be used in compression of various types of images, and thus images created for the needs of smart agricultural production.**

***Index Terms*—SVD; Image compression; Object recognition.**

I. Introduction

THE domain of precision agriculture is reflected in the use of modern information and communication solutions in order to improve productivity by making decisions based on a set of data dating from the past [1]. Recording crops on agricultural land both from the air and from the ground is one of the most commonly used methods for data collection. In this way, depending on the hardware equipment and skills of the operator, photos and videos of different quality can be obtained. As information extraction depends on the quality of the image, recorded images often have to be processed. In addition, high-resolution images require a large amount of storage space, and it is often necessary to compress the image using various digital image processing methods.

Since the digital image data can be represented in the matrix form, the digital image processing methods can utilize a number of mathematical techniques. The essential subject areas are computational linear algebra, integral transforms, statistics and other techniques of numerical analysis. Various digital image processing algorithms can be written in term of matrix equation, hence, computational method in linear algebra come to be an important aspect of the subject [2]. Digital image processing encompasses a wide and varied field of application, such as area of image operation and compression, computer vision, and image analysis (called image understanding). There is the consideration of three types of computerized processing: low level processing where both its inputs and outputs are images; mid-level processing where inputs are images, but outputs are attributes extracted from those images, and higher-level processing that involves "making sense" of an ensemble of recognized objects as in image analysis, and performing the cognitive function associated with human vision [2].

One of the methods that can be used in digital image processing is Singular Value Decomposition (SVD), and this method represents a highlight of linear algebra [3]. More precisely in linear algebra, the SVD of a matrix is a factorization of that matrix into three matrices. It has some interesting algebraic properties and conveys important geometrical and theoretical insights about linear transformations. It also has some important applications in data science. The technique of SVD has a long and somewhat surprising history. It started out in the social sciences with intelligence testing. SVD is known under many different names. In the early days it was called "factor analysis." Other terms include principal component (PC) decomposition and empirical orthogonal function (EOF) analysis. All these are mathematically equivalent, although the way they are treated in the literature is often quite different [4]. Today, singular value decomposition has spread through many branches of science, in particular psychology and sociology, climate and atmospheric science, image processing and astronomy. It is also extremely useful in machine learning and in both descriptive and predictive statistics. From the aspect of digital image processing, image features can be divided in four main groups. Those four groups are visual features, statistical pixel features, transform coefficient features, and algebraic features. Based on those four groups, SVD technique can be observed as an algebraic feature.

The paper is organized as fallows. Second section represents literature review. Third section represents theory of SVD. Fourth section represents application of SVD in image processing, and the fifth section represents main conclusions.

Miloš Ilić is with Faculty of Technical Science, University of Priština temporarily stated in Kosovska Mitrovica, 7 Kneza Miloša, 38220 Kosovska Mitrovica, Serbia (e-mail: milos.ilic@pr.ac.rs).

Petar Spalević is with Faculty of Technical Science, University of Priština temporarily stated in Kosovska Mitrovica, 7 Kneza Miloša, 38220 Kosovska Mitrovica, Serbia (e-mail: petar.spalevic@pr.ac.rs).

Dragana Stojnev is with Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18115 Niš, Serbia (e-mail: dragana.stojnev@elfak.rs)

Nebojša Stojanović is with Academy of Southern Serbia, Partizanska 7, 16000 Leskovac, Serbia (nebojsamstojanovic@gmail.com)

Aleksandra Stojnev Ilić is with Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18115 Niš, Serbia (e-mail: aleksandra.stojnev@elfak.ni.ac.rs)

II. Literature Review

Research in the field of application of SVD method in image processing began in the 70's of the 20th century [5,6]. In one of the researches authors reviewed potential application of SVD in various aspects of digital image processing. They conclude that the easiest way to understand the use of SVD is to realize that a sampled image is nothing more than an array of scalar values and therefore an equivalent to a matrix. The theory of SVD is that of representing matrices (and therefore images) as sums of orthogonal matrices of rank one (outer products) [7]. The decomposition of SVD of an image into a space with diagonal representation then allows implementation of simple enhancement procedures using scalar linear or nonlinear filtering. Such filters suggest the existence of a family of images diagonal in the space of eigenimages of the original picture.

Authors in [8] had applied theory of linear algebra to digital image processing. The aim of the research was the application of SVD within two areas of digital image processing: image compression and face recognition. These two specific areas of digital image processing were firstly investigated and then the application of SVD in these image processing areas is tested. Various experiments with different singular value were performed, and the compression result was evaluated by compression ratio and quality measurement. To perform face recognition with SVD, authors treated the set of known faces as vectors in a subspace, called “face space”, spanned by a small group of “base-faces”. The projection of a new image onto the base-face was then compared with the set of known faces to identify the face. Authors used MATLAB as computing environment and programming language for the purpose of implementation and execution of all tests and experiments. Based on the theory and result of experiments, authors found that SVD is a stable and effective method to split the system into a set of linearly independent components, where each of them is carrying own data (information) to contribute to the system, Thus, both rank of the problem and subspace orientation can be determined. Overall, the SVD approach is robust, simple, easy and fast to implement. It works well in a constrained environment. It provides a practical solution to image compression and recognition problem. Image compression was the topic of interest in one more research. Beside image compression, authors in this research illustrate the use of SVD on matrix completion [9]. The former was to convert the original full-rank pixel matrix to a well-approximated low-rank matrix and thus dramatically save the space. After that, authors recover a pixel matrix with a large number of missing entries by using nuclear norm minimization, in which some singular value thresholding algorithm is used. For both applications, authors conduct numerical experiments to show the performance and point out some possible improvements in the future. Authors conclude that SVD used in image compression process effectively reduces the size of bitmap images. Reducing the image size leads to significant savings in memory space. As a result, the difference between the image restored from the approximated SVD and the original one is negligible and undistinguishable by human eyes. The second application was created in order to recover digital images in the absence of data due to various reasons. The recovery of the full data set from the observed ones largely resembles other related challenges. In order to reduce the use of computing power, authors used algorithm that first identifies the active subspaces and subsequently applies SVD to a matrix that is much smaller in size for each iteration. The authors managed to reconstruct the image with as many as 75% of missing data points.

Authors in [10] show a novel technique for wavelet-based corner detection using singular value decomposition. In presented approach, SVD facilitates the selection of global natural scale in discrete wavelet transform. Authors define natural scale as the level associated with most prominent (dominant) eigenvalue. Created eigenvector corresponding to dominant eigenvalue is considered as the natural scale. The corners are detected at the locations corresponding to modulus maxima. This technique has the advantage of analyzing the wavelet decomposition at natural scale. Algorithm is given for the selection of natural wavelet scale under discrete wavelet domain.

Authors in [11] describe one of the approaches, which integrate both singular value decomposition of each image to increase the compactness density distribution and hybrid color space suitable to this case constituted by the three relevant chromatics levels deduced by histogram analysis. More precisely, their proposition describes the efficiency of SVD and color information to subtract background pixels corresponding to shadows pixels. Singular value decomposition is used in order to increase the compactness power of each distribution. In the same time SVD approximate a new image, which is manipulated, and determine a suitable color space constituted by significant levels among set of levels commonly used in color image analysis. Furthermore, this technique proves that the gray levels and the RGB space are not efficient for all applications. However, the deduced color space (Hrb) shows convincing results and confirm the efficiency of this method.

III. Theory Of Singular Value Decomposition

In this section, a mathematical description of the SVD method is given which is needed in order to understand what this method represents, its implementation and its application to image compression problems.

The decomposition of the matrix into singular values is a factorization that occurs as a step in many algorithms of applied linear algebra. It is equally important in a conceptual sense because it describes the properties of a factorized matrix. Singular values of a matrix represent a generalization of the concept of eigenvalues of square matrices and exist for an arbitrary square or rectangular matrix. This type of factorization overcomes the problems encountered in the process of diagonalization of square matrices, and on that occasion retains the most important properties of orthogonal diagonalization of symmetric matrices [12]

The starting point of the SVD procedure is that an arbitrary rectangular matrix $A \in M_{mxn}$ represents a linear mapping of an n-dimensional vector space into an m-dimensional space $A: R^n \rightarrow R^m$, or $A: C^n \rightarrow C^m$ in a complex case. The diagonalization of square matrices is related to the choice of one base of vector space that facilitates calculations. Rectangular matrices act over different spaces. It is a natural question whether two orthonormal bases of these two vector spaces can be found, which will describe its properties as transformations through the simplest form of matrix *A*.

A matrix of size $m \times n$ is a grid of real numbers consisting of m rows and n columns. When we have an (m × n)-matrix A and a (n × k)-matrix B, we can compute the product AB which is an (m × k)-matrix. The mapping corresponding to AB is exactly the composition of the mappings corresponding to A and B respectively. Singular Value Decomposition (SVD) states that every (m × n)-matrix A can be written as a product. a graphical representation of the creation of the matrix a is shown in Fig. 1.

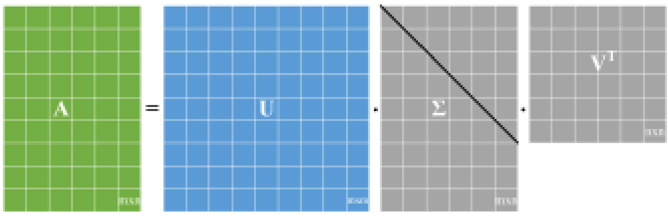


Fig. 1. Construction of matrix A

In this product $U \in M_{mxm}$ and $V \in M_{nxn}$ are orthogonal matrices and the matrix $\Sigma \in M_{mxn}$ consists of descending non-negative values on its diagonal and zeros elsewhere, so it is valid

$$A = U\Sigma V^T \qquad (1)$$

The diagonal elements σ_i of the matrix Σ are called the singular values of the matrix A. The matrix Σ can be shown in following:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_r & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \sigma_{r+1} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \sigma_n \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix}. \qquad (2)$$

The columns of the matrix U are called the left singular vectors, and the columns of the matrix V are the right singular vectors. It is a consequence of equality, as in

$$AV = U\Sigma \text{ and } U^T A = \Sigma V^T \qquad (3)$$

$$Av_i = \sigma_i u_i \quad u_i^T A = \sigma_i v_i^T \quad i = 1,2,\ldots,\min\{m,n\} \qquad (4)$$

Equation 1 can be viewed as a transformation of a unit sphere of n-dimensional space. If viewed in this way, this equality can be analyzed through a series of steps. The use of the orthogonal matrix V^T is based primarily on the rotation of the unit sphere of *n*-dimensional space by rotating the natural base into the base of the principal axes v_i. The diagonal matrix *Σ* stretches or contracts the sphere along the major axes into the ellipsoid. At the same time, a rectangular diagonal matrix Σ incorporates a sphere from n − dimensional space into an ellipsoid in m − dimensional vector space. The orthogonal matrix *U* rotates the ellipsoid to new major axes u_j in to m − dimensional space. The dimension of the space through which the ellipsoid extends is equal to the number of nonzero diagonal values of the matrix Σ [12].

There is a neat way to remove U and see V by itself. Multiply A^T times A. In this case practically SV decomposition matrix is closely related to the Gram matrix A^TA, and the equation was obtained as in

$$A^T A = (U\Sigma V^T)^T (U\Sigma V^T) = V\Sigma^T \Sigma V^T \qquad (4)$$

U^TU disappears because it equals I. Multiplying those diagonal Σ^T and Σ gives σ_i^2 values. That leaves an ordinal diagonalization of the crucial symmetric matrix A^TA, whose eigenvalues are σ_i^2 values. The diagonal matrix $\Sigma^T\Sigma$ is orthogonally similar to the Gram matrix A^TA [12]. Therefore, the eigenvalues σ_i^2 of the matrix $\Sigma^T\Sigma$ are at the same time the eigenvalues of the matrix A^TA [13]. There are many properties and attributes of SVD. Some of the most important SVD properties and attributes in terms of application in the field of image processing are numbered bellow.

- singular values of $\sigma_1, \sigma_2, \ldots, \sigma_n$ are uniquely defined, but the matrices *U* and *V* are not,
- since $A^TA = V\Sigma^T\Sigma V^T$, so *V* diagonalizes A^TA , it follows that the v_j s are the eigenvector of A^TA,
- since $AA^T = U\Sigma\Sigma^TU^T$, so it follows that *U* diagonalizes AA^T and that the u_i 's are the eigenvectors of AA^T
- if *A* has rank of *r* then v_j, $v_j, \ldots,$ v_r form an orthonormal basis for range space of A^T, $R(A^T)$, and u_j, u_j, …, u_r form an orthonormal basis for range space *A*, *R(A)*.
- the rank of matrix *A* is equal to the number of its nonzero singular values,
- the eigenvalues of the A^TA matrix are also the squared singular values of the matrix A.

IV. Application Of SVD In Agricultural Image Compression

SVD can be applied to multiple problems in the domain of digital image processing. Observed from the angle of processing of images that represent agricultural areas, the challenge of image compression stands out. The need for image compression stems from the fact that most images that are obtained both from the air and from the ground have high-resolution, so they take up a large amount of memory. This is especially noticeable if a large number of images need to be stored for a long period in order to create a database of images. As part of the conducted research, the application of SVD practically represents the pre-processing of images obtained by recording agricultural plantations from the air. Initially, the images are collected for the development of a system for the recognition of the occurrence of diseases and pests and predicting the time of application of agrotechnical and chemical measures. As this task requires a significant number of high-resolution images, there is a need to compress them before storing them for further use. In addition to reducing memory usage, the use of compressed images further simplifies the process of their processing and use. The idea was that each of the selected images intended for use in the performed experiment should be represented in the form of a matrix, which can be done with a mathematical approach. Practically observed from the angle of mathematics and possible representation of the digital image, each digital image can be represented as matrix of pixel values.

If black and white images are observed, each little image element or “pixel” has a gray scale number between black and white. In the case of color pictures it has three numbers. Based on that, image compression deals with the problem of reducing the amount of data required to represent a digital image. When an image is SVD transformed, it is not compressed, but the data take a form in which the first singular value has a great amount of the image information. This property allows that only a few singular values need to be used in order to represent the image with little differences from the original.

Image compression using SVD was implemented within the MATLAB software package based on the following procedure. Decomposition of matrix A to the *UΣV^T^* product represents an approximation of the matrix *A* using much smaller values compared to the original matrix.

$$A=\sigma_1 u_1 v_1^T+\sigma_2 u_2 v_2^T+\ldots+\sigma_r u_r v_r^T+0u_{r+1}v_{r+1}^T+\ldots \quad (5)$$

As singular values are always greater than zero, adding dependent members, where singular values are equal to zero, has no effect on the image. In the end, we get the equation with the members:

$$A=\sigma_1 u_1 v_1^T+\sigma_2 u_2 v_2^T+\ldots+\sigma_r u_r v_r^T \quad (6)$$

Furthermore, the obtained matrix can be further approximated by omitting the singular members of matrix A. As the singular values are sorted in descending order, the last members have the least influence on the final image. This is how the size of the memory space needed to store the newly created image is reduced. The closest matrix of rank *k* is obtained by truncating those sums after the first *k* terms:

$$A=\sigma_1 u_1 v_1^T+\sigma_2 u_2 v_2^T+\ldots+\sigma_k u_k v_k^T \quad (7)$$

If we observe the compression ratio, it can be calculated as in:

$$R=\frac{nk+k+mk}{nm}*100 \quad (8)$$

Where *R* is the compression percentage, *k* is the chosen rank for truncation; *m* and *n* are the number of rows and columns in the image respectively. The calculation of the compression coefficient as well as the size of the compressed image is shown based on one selected representative image. This image was taken over the field with the use of a drone. Compression results obtained after the execution are displayed in the appendix on the Fig 3. The resolution of the original image is 3840 x 2160, the number of pixels of the uncompressed image is 8294400, where 3840 represents the vertical and number 2160 the horizontal number of pixels of the original image. If k = 20 which can be viewed as the number of iterations equation 7 will look like:

$$A=\sigma_1 u_1 v_1^T+\sigma_2 u_2 v_2^T+\ldots+\sigma_{20} u_{20} v_{20}^T \quad (9)$$

Based on the previous equation each of u_i contains 648 components, each of v_i contains 480 components, and how each σ_i is the scalar that represents 1 component. If we denote the compressed image with N_K, the number of pixels of the compressed image can be obtained as in:

$$N_k=20*(3840+2160+1)=120020 \quad (10)$$

The compression ratio of image can be calculated as follows:

$$R=\frac{2160*20+20+3840*20}{2160*3840}*100=1.45\% \quad (11)$$

Finally, after 20 iterations we got approximately the same image, which is 1.45% of the compressed image, while after 150 iterations we got a good compressed image, which is 10.85%. Compared to a compressed image with 20 iterations, an image with 150 iterations looks visually much better with noticeable sharpness and detail while still taking up little storage space. In the Fig. 2 error distribution between the compressed and the original image, which is taken as an example, is shown.

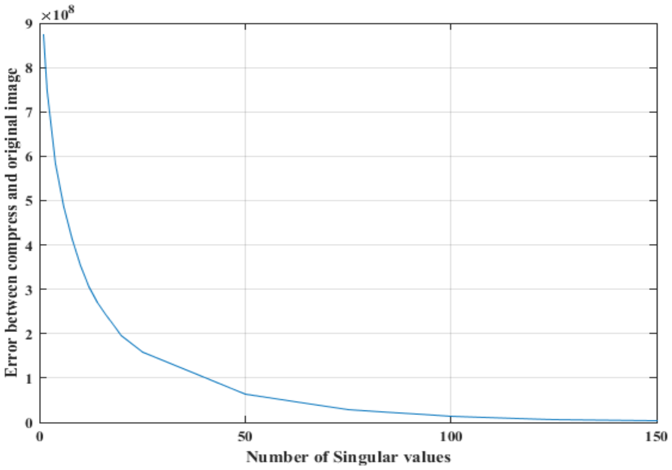


Fig. 2. Error distribution in compression process of selected image

As it can be seen from Fig. 2, as the number of singular values increases, the error decreases. When working with black and white images, experimental results have shown that less iteration is required in order to obtain adequate quality of compressed image.

If we look at the images given in Fig. 3, it is noticeable that with an initially small number of iterations, the quality of the compressed image decreases. When more than 75 singular values are used, the quality of the compressed image is almost identical to the original image, while the size of the compressed image is reduced by half compared to the original. As these are images whose purpose is to be used in the decision-making process, it is necessary for the image quality to be appropriate, which corresponds to a larger number of singular values.

The original image takes up 5.43MB of space, while the space required to store compressed images ranges from 888KB for an image obtained using 20 singular values to 1.35MB for an image obtained using 150 singular values.

V. Conclusion

Innovations in the field of digital image creation have created the possibility of creating high-resolution images. These images represent a significant source of information. However, high-resolution images take up significant memory space. If it is a large number of digital images, as is the case with images created for the needs of agriculture, where large areas are photographed by dividing the pre-defined quarters, it is necessary to process and compress the images obtained in this way. Due to the importance of information, compression must not reduce the quality of the image itself.

In this paper, we presented the fact that mathematical methods like SVD can be applied to image compression by looking at each of its pixels. With the use of this method we were able to compress both black and white and color images, obtained by aerial photography. In this way, the pre-processing of the images needed within the larger system for monitoring agricultural land was performed, which enabled easier storage and use of these images.

In addition to the use of both black and white, and color images in compression, the detection domain of the same or similar objects can also be used. This application can be important in the field of monitoring the situation on agricultural land. The use of SVD in object detection allows the comparison of the photographed object with pre-selected objects within the database. It is practically necessary to create a database of objects based on which the comparison will be made. Applied to the domain of agriculture, SVD can be used in estimating the number of weed communities on agricultural land, detection and marking of areas in fruit crops, detection and marking of uncultivated land, forests, underwater areas, etc. The starting point for detecting objects using SVD are certainly quality digital images. The implementation of the application of SVD in order to compare objects on compressed images in the database and new objects on newly obtained images is an idea for future research. This method can provide detection of pathogens and pests on sown crops.

Appendix

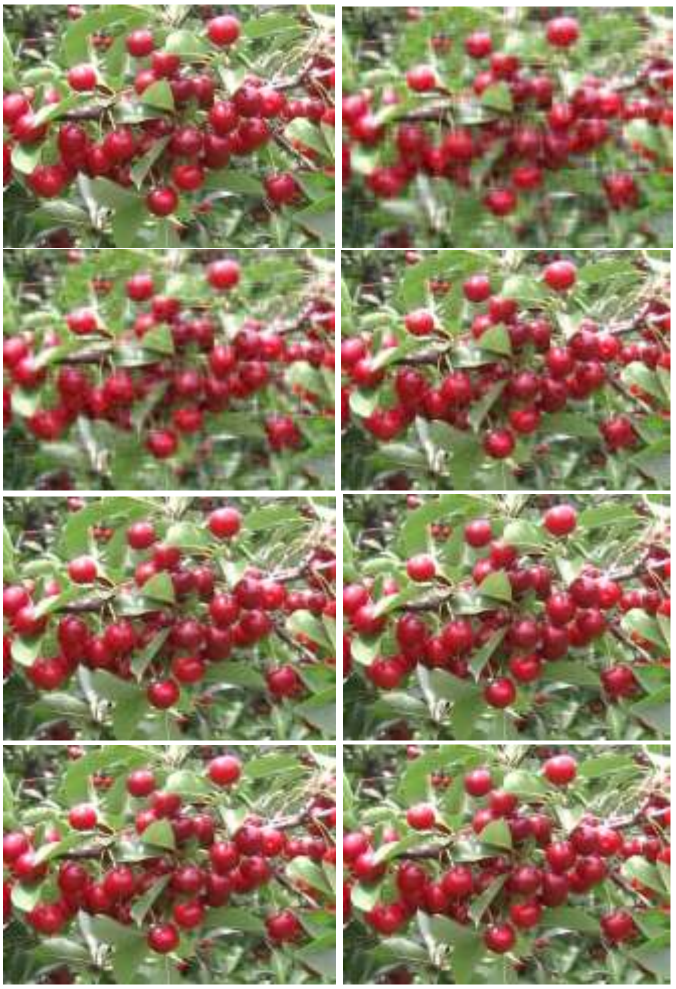


Fig. 3. Example of image compression with singular values used {(original image, 20), (25,50), (75,100), (125, 150)} respectively

ACKNOWLEDGMENT

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia in the projects III-43007, TR 35026 and TR 32023

REFERENCES

[1] Z. Qin , “Precision agriculture technology for crop farming,” NW, US, CRC Press Taylor & Francis Group, 2016.

[2] J. Bernd, “*Digital Image Processing*”, Berlin, Germany, Springer-Verlag Berlin Heidelberg, 2005.

[3] R. A. Sadek, “SVD Based Image Processing Applications: State of The Art, Contributions and Research Challenges,” *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 7, pp. 26-34, 2012.

[4] Y. Wang, T. Tan, Y. Zho, “Face Verification Based on Singular Value Decomposition and Radial Basis Function Neural Network,” National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, 2000.

[5] G. H. Golub, C. Reinsch, “Singular value decomposition and leasts quaress solutions,” in *Linear Algebra. Handbook for Automatic Computation*, Berlin, Germany, Springer, Berlin, Heidelberg, 1971, ch 2, pp. 03-420.

[6] T. S. Huang, W. F. Schrieber, 0. J. Tretiak, “Image processing,” Proc. IEEE, vol. 59, pp. 1586-1609, Nov. 1971.

[7] H.C. Andrews, C.L. Patterson, “Singular Value Decompositions and Digital Image Processing,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 1, pp. 26-53, February, 1976.

[8] C. Lijie, “Singular value decomposition applied to digital image processing,” Division of Computing Studies, Arizona State University Polytechnic Campus, Mesa, Arizona State University polytechnic Campus, pp. 1-15, 2006.

[9] Z. Chen, “Singular Value Decomposition and its Applications in Image Processing,” ICoMS 2018: Proceedings of the 2018 International Conference on Mathematics and Statistics, Porto Portugal, pp. 16-22, 2018.

[10] A. Quddus, M. Gabbouj, “Wavelet based corner detection using singular value decomposition,” International Conference Acoustics, Speech, and Signal Processing, Istanbul, Turkey, Turkey, 2000.

[11] M. M. Jlassi, A. Douik, H. Messaoud, “Objects Detection by Singular Value Decomposition Technique in Hybrid Color Space: Application to Football Images”, *International journal of computers communications & control*, vol. 5, no. 2, pp. 193-204, June, 2010.

[12] J. Dzunic, “*Matrix methods - Applications through Python*,” Nis, Serbia, University of Nis, Faculty of Electronic engineering, 2019.

[13] G. Strang, “Introduction in Linear Algebra,” 4th ed. USA, Massachusetts Institute of Technology Wellesley - Cambridge Press, 2009.

Inkrementalni razvoj 3D video-igre *Arena* kroz praktičan rad studenata

Vojislav Bogosavljević i Igor Tartalja

Apstrakt—**U radu je opisano iskustvo inkrementalnog razvoja 3D video-igre kroz njenu evoluciju od skeleta koda prikazanog studentima na pokaznoj laboratorijskoj vežbi na predmetu Računarska grafika, preko kontrolne laboratorijske vežbe i kasnijeg domaćeg zadatka (projekta), sve do diplomskog rada studenta. Cilj razvijene igre je da se za ograničeno vreme sakupi što veći broj poena i preživi u areni sa nekoliko prostorija, izbegavanjem prepreka, uz sakupljanje različitih bonusa. Softver je modelovan na standardnom jeziku UML (*Unified Modeling Language*), a implementiran na programskom jeziku *Java*, uz korišćenje bibilioteke *JavaFX*.**

Ključne reči—**računarska grafika; 3D video-igra; inkrementalni razvoj softvera; laboratorijska vežba; JavaFX.**

I. Uvod

Računarska grafika je nauka i umetnost komunikacije vizuelnim putem pomoću računarskog prikazivača i uređaja za interakciju [1]. Njen cilj je sinteza statičke ili pokretne slike uz interakciju sa korisnikom. Računarska grafika prožima skoro sve oblasti ljudske delatnosti, a izvesno je da će, kako čovečanstvo dalje bude napredovalo u tehnološkom smislu, oslanjanje na računarsku grafiku biti još veće. Pored značajnih domena primene, poput nauke, tehnologije, medicine, obrazovanja, umetnosti i drugih, njena primena koja daje ovoj oblasti i najjači pogon za razvoj je industrija igara. U ovom radu obrađen je inkrementalni razvoj jedne relativno jednostavne 3D video-igre.

Na Elektrotehničnom fakultetu Univerziteta u Beogradu jedan od izbornih predmeta jeste *Računarska grafika*. U okviru tog predmeta se izučavaju osnovni koncepti i dobija teorijska podloga, a takođe se stečena znanja primenjuju i utemeljuju kroz praktične obaveze na predmetu. Ovaj rad predstavlja jednu studiju slučaja inkrementalnog razvoja 3D video-igre kroz praktične obaveze studenata na ovom predmetu.

Na predmetu *Računarska grafika* se za realizaciju praktičnih zadataka koristi programski jezik Java i biblioteka JavaFX, koja omogućava dovoljno jednostavno i efikasno postizanje rezultata u 2D i 3D grafici [2]. Zbog svoje jednostavnosti i prijemčivosti (a bez gubljenja sveobuhvatnosti), prema iskustvima nastavnika na kursu, ovaj jezik sa ovom bibliotekom predstavlja dobar izbor za akademske svrhe. Razvojno oruženje u kom je razvijan softver kroz predispitne obaveze i diplomski rad je integrisano okruženje *NetBeans* [3], koje nudi podršku za *JavaFX*.

U radu je opisan inkrementalni razvoj jedne 3D video-igre kao teme za izradu najpre laboratorijskih vežbi, zatim domaćeg zadatka (projekta) i, na kraju, diplomskog rada. Tema igre je arena sa tri prostorije u kojima igrač nastoji da preživi određeno vreme u borbi sa različitim vrstama prepreka u različitim prostorijama, kako što su šiljci koji iskaču iz tla, projektili koji preleću kroz prostoriju, odnosno sečiva koja padaju sa plafona.

Rad je prvenstveno namenjen čitaocima iz akademskih ustanova, ali opisana iskustva mogu biti od koristi i mlađim razvojnim programerima u industriji igara.

U sledećem poglavlju se opisuje rešavani problem. U trećem poglavlju je dat pregled igara koje su srodne igri *Arena* ili su poslužile kao inspiracija za nju. U četvrtom je sažeta funkcionalna specifikacija kompletne igre. U petom je dat kratak pregled najvažnijih delova arhitekture programa. U šestom poglavlju je opisana implementacija igre, prikazane su tehničke karakteristike, kao i nekoliko interesantnih detalja implementacije. Sedmo poglavlje posvećeno je diskusiji rezultata. U poslednjem, osmom, poglavlju dat je zaključak, sa nekim smernicama za mogući dalji razvoj igre.

II. Problem

Znanja usvojena na predmetu *Računarska grafika* studenti imaju priliku da praktično primene kroz laboratorijske vežbe, zatim domaći zadatak (projekat) i na kraju kroz diplomski ili master rad. Organizuju se četiri vežbe, od kojih su prva i treća pokaznog karaktera, a druga i četvrta kontrolnog; prve dve su vezane za 2D, druge dve za 3D grafiku. Polovinu poena na predmetu studenti stiču kroz kontrolne laboratorijske vežbe i/ili domaći zadatak

Na pokaznim laboratorijskim vežbama studenti se upoznaju sa programom (skeletom koda) koji će se na kontrolnoj vežbi nadograđivati za ocenu. Skelet koda predstavlja jezgro igre, te sadrži osnovne funkcionalnosti objedinjene u smislenu celinu. On usmerava kasnije projektovanje igre, obezbeđujući pri tome visok stepen fleksibilnosti u implementaciji zadataka vezanih za grafičke objekte, njihovu animaciju i interakciju sa korisnikom. Studenti taj skelet proširuju dodatnim funkcionalnostima i kroz to utvrđuju svoja znanja, pokazuju u kojoj su meri savladali materiju i vežbaju da vrše izmene, odnosno proširenja nad tuđim kodom. Na 3. laboratorijskoj vežbi se studenti upoznaju sa skeletom koda i ostavi im se oko 30min za implementaciju neke dodatne funkcionalnosti,

Vojislav Bogosavljević – Elektrotehnički fakultet, Univerzitet u Beogradu, Bulevar Kralja Aleksandra 73, 11020 Beograd, Srbija (e-mail: vbogos3@gmail.com).

Igor Tartalja – Elektrotehnički fakultet, Univerzitet u Beogradu, Bulevar Kralja Aleksandra 73, 11020 Beograd, Srbija (e-mail: tartalja@etf.rs).

eventualno uz pomoć demonstratora, što se ne ocenjuje. Na laboratorijskoj vežbi studenti dobijaju veći broj (do 20) zadataka za dodavanje novih funkcionalnosti skeletu koda.

Kao tema treće i četvrte laboratorijske vežbe, prikazanih u ovom radu, odabrana je 3D video-igra. Motivacija za to je činjenica da problem programiranja video-igre obuhvata više različitih aspekata. Najpre, što je od naročitog interesa za predmet *Računarska grafika*, tu su vizuelni dizajn i realizacija grafičkog dela, a zatim osmišljavanje igre i iskustva koje ona pruža (eng. *gameplay*). Dalje, tu je primena postojećih i razvoj novih algoritama za implementaciju delova igre, pisanje kvalitetno strukturiranog, održivog i proširivog koda, te dizajn zvuka (pozadinske muzike i zvučnih efekata) i drugo [4]. Značajna je činjenica da se ishodi razvoja veoma brzo uočavaju, odnosno potrebno je relativno malo vremena i truda da bi se postigli vidljivi rezultati, što može biti veoma motivišuće studentima.

Praktični deo predmeta Računarska grafika se kompletira kroz domaći zadatak (projekat). Ovaj projekat obuhvata dalju nadogradnju implementacije sa laboratorijske vežbe, a projektni zahtevi su osetno složeniji i obimniji. Na kraju, polazeći od nivoa domaćeg zadatka, igra se može dodatno proširiti još kompleksnijim funkcionalnostima, kako bi dostigla nivo diplomskog ili master rada.

III. Srodna rešenja

Arena je žanrovski najbliža opštoj kategoriji akcionih igara (eng. *action game*) u kojima je naglasak na fizičkim izazovima, što uključuje koordinaciju između ruke i oka i brzinu reakcije [5]. Mogla bi se povući i paralela sa platformskim igrama (eng. *platformer*) kao podžanrom akcionih, u kojima je cilj, u opštem slučaju, skakati i penjati se do odvojenih platformi, uz izbegavanje prepreka [6]. Međutim, čest je slučaj da vizura platformske igre bude iz perspektive trećeg lica, što ovde nije slučaj. Perspektiva u igri *Arena* je (primarno) iz prvog lica (eng. *first-person*), rađena najpre po ugledu na “pucačke” igre iz prvog lica [7]. Takođe, ne postoji mogućnost skakanja, pošto se radnja igre odigrava u jednoj horizontalnoj ravni.

Arena nije rađena kao klon neke postojeće igre, ali pojedini aspekti su realizovani po uzoru na druge igre. Igre iz kojih je stigla određena inspiracija navedene su u nastavku.

A. Rayman 2: The Great Escape

Rayman 2: The Great Escape [8] je platformska video-igra u kojoj igrač kontroliše Rejmena, humanoidnog lika karakterističnog po tome što mu šake, stopala i glava lebde odvojeni od torzoa. On ima mogućnost gađanja protivnika energetskim projektilima, a dodatnu kontrolu kretanja prilikom skokova mu dozvoljava pokretanje kose u maniru elise helikoptera. Radnja igre se kreće oko invazije robota-gusara iz svemira koji nameravaju da okupiraju svet. Igru karakterišu kvalitetno dizajnirani nivoi, sveprisutna mistična atmosfera i upečatljivi likovi. Elementi igre koji su poslužili kao inspiracija za *Arenu* jesu raznovrsne prepreke, poput pipaka koji iskaču iz zidova i projektila kojima protivnici gađaju igrača.

B. Croc 2

Croc 2 [9] je platformska video-igra u kojoj igrač kontroliše Kroka, krokodila u potrazi za nestalim roditeljima. Igrač ima razne mogućnosti kontrole, kao što su višestruki skok i različite vrste napada. Svet je dosta otvoren i često je na igraču da odabere kojim će redosledom prelaziti postave. Za ovu igru je karakterističan veliki broj raznovrsnih nivoa i njihov konceptualni i vizuelni dizajn, od kojih su neki elementi poslužili kao uzor za *Arenu*, među kojima su prepreke poput iskačućih šiljaka i srca za sakupljanje, radi obnavljanja izgubljenih života.

C. Half-Life

U pitanju je jedna od najuticajnijih igara iz žanra pucačkih igara iz prvog lica, koja se ubraja među najbolje igre svih vremena [10]. Srž scenarija igre je borba protiv vanzemaljskih i ljudskih protivnika i rešavanje zagonetki. Igrač kontroliše Gordona Frimena, naučnika koji se bori za svoj život nakon što u istraživačkom centru eksperiment sa vanzemaljskim materijalom pođe naopako. Iako *Arena* žanrovski nema pucački karakter, generalni osećaj kontrole iz prvog lica i elementi vizuelnog identiteta (na primer, veličina vidnog polja, brzina kretanja i dizajn šiljaka i sečiva) crpeli su inspiraciju iz igre *Half-Life*; valja naglasiti da ova igra poseduje i platformske elemente.

IV. Funkcionalnosti softvera

Arena je 3D akciona igra u kojoj je cilj igrača da sakupi što više poena za ograničeno vreme, uz izbegavanje prepreka. Iako je primarno igra iz prvog lica, postoji i druga kamera u centru prostorije na vertikalnoj osi, što omogućava pregled svih delova prostorije i pogled na avatara igrača. Glavni meni omogućava korisniku da pokrene igru, vidi tabelu najboljih rezultata, uđe u podešavanja i napusti program. Pre započinjanja igre neophodno je uneti ime igrača.

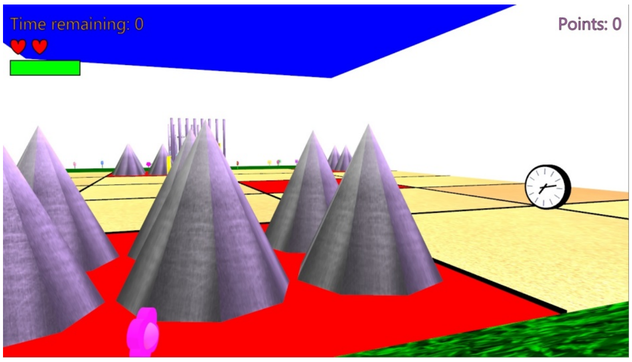
Igrač upravlja kretanjem svog avatara komandama sa tastature, uključujući mogućnosti hodanja ili trčanja. Pomeranjem miša se usmerava kamera iz prvog lica, odnosno menja orijentacija igrača. Igru je moguće privremeno zaustaviti (pauzirati) i nastaviti.

Arena je podeljena na tri prostorije, sa različitim vrstama skrivenih prepreka u svakoj: šiljci (Sl. 1), projektili (Sl. 2) i sečiva (Sl. 3). Pri kontaktu sa njima, igrač gubi jedan život. Tlo prostorija je podeljeno na ploče. Nešto ranije u odnosu na pojavljivanje prepreke, odgovarajuća ploča (u 1. prostoriji), red ploča (u 2. prostoriji) ili grupa susednih ploča (u 3. prostoriji) na kojima će se pojaviti prepreka će pocrveneti sugerišući igraču nastupajuću opasnost. Nad poljima se stvaraju predmeti za sakupljanje: novčići koji daju poene, srca koja obnavljaju živote i satovi koji daju dodatno vreme (na slikama prostorija mogu se videti i ti predmeti).

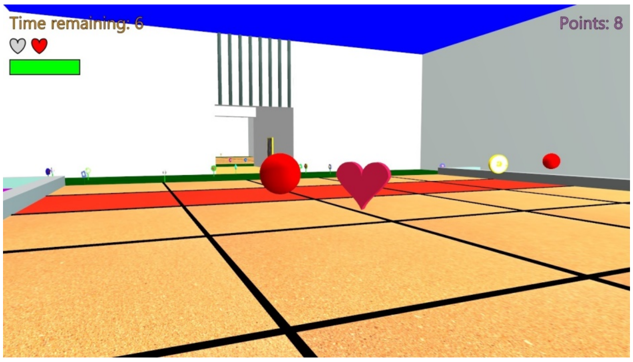
Igračev interfejs (eng. *Head-up Display*, HUD), stalno prikazan tokom igre u vrhu ekrana, sadrži informacije o preostalom vremenu, životima i energiji za trčanje. Energija se troši dok igrač trči, a dok stoji ili hoda ona se obnavlja.

Odbrojavanje vremena u prostoriji počinje čim igrač zakorači u prostoriju iz hodnika koji joj prethodi. Nakon isteka vremena u jednoj prostoriji, otvaraju se kapije hodnika ka narednoj.

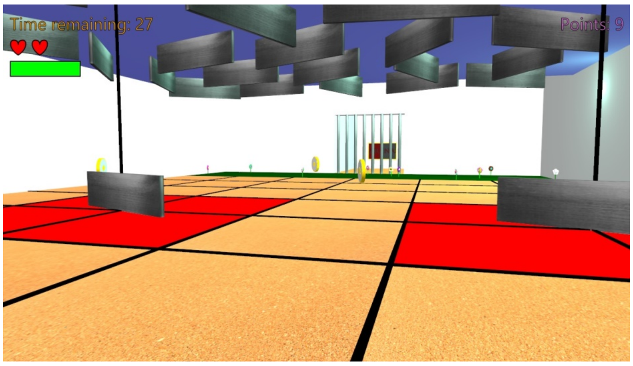
Igra se uspešno završava ukoliko igrač dočeka istek vremena u poslednjoj prostoriji, a neuspešno ako izgubi sve živote. Bolje je plasiran onaj igrač koji je sakupio više poena. Detalji igre opisani su u dokumentu Prilog A (videti Dodatak).



Sl. 1. Prva prostorija sa preprekama – šiljcima



Sl. 2. Druga prostorija sa preprekama – projektilima

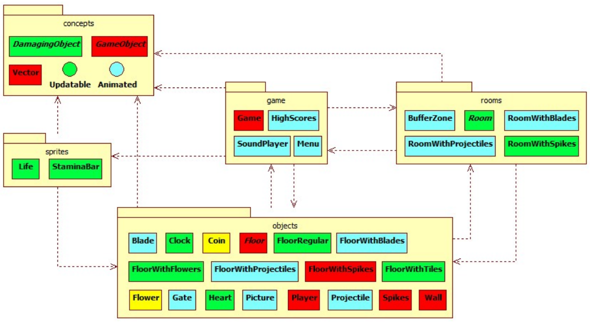


Sl. 3. Treća prostorija sa preprekama – sečivima

V. PROJEKAT SOFTVERA

Na Sl. 4 prikazan je UML dijagram paketa sa svim klasama i interfejsima u projektu. Crvenom bojom označene su klase iz skeleta koda, odnosno sa pokazne vežbe, žutom klase dodate na kontrolnoj laboratorijskoj vežbi, zelenom klase dodate kroz domaći zadatak, a tirkiznom klase dodate kroz diplomski rad. Detalji projekta softvera opisani su u dokumentu Prilog B (videti Dodatak).

U paketu `concepts` nalaze se klase i interfejsi koji su osnova za veliki broj ostalih klasa: `GameObject` je natklasa svih klasa u paketima `objects` i `rooms`, `Vector` je prisutan kao polje u klasi `GameObject` i kao parametar njenog konstruktora, dok interfejse `Updatable` i `Animated` implementiraju mnoge klase iz navedenih paketa, kao i paketa `sprites`.



Sl. 4. UML dijagram paketa razvijenog softvera

Klase koje predstavljaju opštije celine i pokretače dešavanja u igri nalaze se u paketu `game`. Među njima je glavna klasa `Game` u kojoj je i `main()` metod. Ta klasa je ključna za tok igre, omogućava pauziranje i prelazak u meni, a takođe poseduje tajmer na čije se otkucaje odigravaju mnoge važne stvari, poput izazivanja događaja u određenim trenucima, detekcije kolizija, računanje protoka vremena i slično. Praktično povezuje sve klase u projektu.

Paket `objects` sadrži 3D objekte koji se pojavljuju u igri, kao što su satovi, tlo, zidovi i slično. Tu je i klasa `Player` koja ima nešto složenije mogućnosti: predstavlja igrača, ima nosač za kameru za pogled iz prvog lica, rukovaoce događajima za korisnički ulaz radi kontrole avatara i vrši značajan deo upravljanja kretanjem po areni.

Iako predstavljaju 3D objekte, odnosno prostorije u areni, klase unutar paketa `rooms` su u njega izdvojene iz razloga što su u pitanju objekti sastavljeni od većeg broja jednostavnijih, iz paketa `objects`. Takođe, ove klase realizuju funkcionalnosti koje utiču na dešavanja u igri i nisu samo statične celine koje služe da budu korišćene od strane drugih delova koda (kao većina klasa u paketu `objects`).

U `sprites` su smešteni 2D objekti koji su deo korisničkog interfejsa – srca koja predstavljaju preostale živote i pokazivač preostale energije za trčanje.

VI. IMPLEMENTACIJA SOFTVERA

Osnova za implementaciju igre je skelet koda, koji se stavlja studentima na raspolaganje. Polazeći od njega, igra se proširuje do nivoa kontrolne laboratorijske vežbe, na kojoj studenti samostalno realizuju dodatne funkcionalne zahteve. Te izmene su prilagođene ograničenom vremenu izrade laboratorijske vežbe, te su mahom estetske prirode, sa izuzetkom dodavanja novčića, čime se igri nazire poenta. Odatle, realizacijom stavki za domaći zadatak, aplikacija počinje da poprima opšte crte poznate za veliki broj igara: ograničeno vreme za sakupljanje što većeg broja poena i mogućnost gubitka života čine igru izazovnom i otvaraju mogućnost postizanja uporedivih rezultata po odigranim partijama, dok postojanje obnovljivih zaliha energije za trčanje i pojava satova za dodatno vreme navodi igrača da se pametno kreće po areni. Nadogradnja od nivoa domaćeg zadatka do diplomskog rada kompletira video-igru: više prostorija daje veću slobodu kretanja i predstavlja mali svet za sebe, pozadinska muzika i zvučni efekti doprinose atmosferi, a postojanje menija i mogućnost pauziranja igre nagoveštavaju da je u pitanju celovito rešenje, nalik modernim igrama. U nastavku će biti ukratko opisani neki interesantni elementi implementacije.

A. Implementacija pogleda iz prvog lica

Za okretanje kamere (pogleda igrača), pomoću miša, bilo je jednostavno u klasi `Player` realizovati rotaciju u horizontalnoj ravni (pri pomeranju miša levo ili desno), jer je potrebno promeniti samo ugao objekta bibliotečke klase `Rotate` zaduženog za rotaciju u toj ravni. Osa rotacije je podešena da bude Y osa, pošto se igra odvija u jednoj horizontalnoj ravni, a Y osa je normalna na nju. Osa rotacije u vertikalnoj ravni se podešava sledećim kodom:

```
upDownRotation.setAxis(
  new Point3D(
    Math.cos(horizontalAngle*Math.PI/180),
    0,
    -Math.sin(horizontalAngle*Math.PI/180)
  )
);
```

gde je `upDownRotation` objekat tipa `Rotate`, zadužen za rotaciju u vertikalnoj ravni (pri pomeranju miša gore ili dole), a `horizontalAngle` ugao rotacije u horizontalnoj ravni, koji se izračuna nakon pomeranja miša levo ili desno. Tako postavljena osa rotacije predstavlja horizontalnu X osu igrača. Kasnijim pozivom metode `setAngle()` objekta `upDownRotation` kamera se rotira u vertikalnoj ravni za prosleđenu vrednost ugla `verticalAngle` izračunatu nakon pomeranja miša gore ili dole.

B. Implementacija upozoravajućeg bojenja reda ploča

U prostoriji sa projektilima potrebno je da redovi ploča postanu crveni kao upozorenje pred lansiranje projektila. Zbog konstruktora koji su na raspolaganju za bibliotečku klasu `KeyFrame`, bilo je neophodno kao jedan od parametara proslediti svojstva njihovih boja (`ObjectProperty<Color>`) kao niz objekata tipa `KeyValue`. Tu su se korisnim pokazale mogućnosti Java 8 i mogućnosti koje nudi kroz klase iz paketa `java.util.stream` [11]. Pozivanjem statičkih metoda `stream()` klasa koje ga poseduju, odnosno istoimenog nestatičkog metoda objekata koji ga poseduju, vraća se objekat klase koja implementira interfejs `Stream`. Taj interfejs apstrahuje sekvencu elemenata nad kojom se mogu vršiti sekvencijalne i paralelne agregatne funkcije pomoću njegovih metoda, kao što su `toArray()`, `forEach()` i druge. Metod `map()` objekata klasa koje implementiraju interfejs `Stream` svaki element iz sekvence pojedinačno preslikava u neki drugi objekat (potencijalno drugog tipa) funkcijom prosleđenom kao parametar metoda, tako da novonastali objekat tipa `Stream` predstavlja sekvencu tih novih objekata. Sledećim kodom je rešen navedeni problem:

```
Arrays
    .stream(tiles)
    .map(col -> col[zIndex])
    .map(tile -> new KeyValue(
        ((PhongMaterial)tile.getMaterial())
            .diffuseColorProperty(),
        Color.RED,
        Interpolator.LINEAR))
    .toArray(KeyValue[]::new)
```

gde su `tiles` ploče na koje se deli tlo prostorije, tipa `Box[][]`, a `zIndex` indeks reda koji treba da pocrveni. Specificiranjem prvog indeksa dvodimenzionalnog niza `tiles` određuje se kolona, a drugim i pojedinačna ploča u datom redu. Imajući to u vidu, pozivom statičkog metoda `Arrays.stream(tiles)` dobija se sekvenca kolona ploča, a u prvom pozivu `map()` svaka kolona se preslikava u ploču te kolone, i to sa istim indeksom (`zIndex`), što predstavlja red koji treba da pocrveni. Zatim se svaka od tih ploča drugim pozivom metoda `map()` preslikava u sebi odgovarajući nov objekat tipa `KeyValue` tako što se za svaku ploču iz reda poziva konstruktor klase `KeyValue`, kom se kao prvi parametar prosleđuje difuzna boja date ploče, kao drugi crvena boja, a kao treći objekat linearnog interpolatora. Naposletku se dobijena sekvenca objekata tipa `KeyValue` pretvara u niz pozivom metoda `toArray()`.

C. Tehničke karakteristike softvera

U Tabeli 1 prikazane su tehničke karakteristike rešenja igre po različitim fazama razvoja. Za analizu je korišćen program *SourceMonitor* [12].

TABELA I

TEHNIČKE KARAKTERISTIKE IMPLEMENTACIJE IGRE *ARENA* PO FAZAMA

	Skelet koda	Kontrolna lab vežba	Domaći zadatak	Diplomski rad
Broj linija koda	673	1146	1979	4085
Broj fajlova	8	10	21	34
Broj klasa	11	14	25	42
Broj metoda	43	57	106	247
Veličina izvršnog fajla	766 KB	777 KB	795 KB	26,2 MB
Veličina *NetBeans* projekta	2,81 MB	2,86 MB	2,94 MB	81,7 MB

Iz Tabele 1 se može uočiti da je broj linija koda kontrolne laboratorijske vežbe skoro duplo veći od broja linija skeleta koda, a kod domaćeg zadatka taj broj je približno utrostručen. Na nivou diplomskog rada ima više nego duplo linija koda u odnosu na domaći zadatak. Nesrazmeran skok u veličini izvršnog fajla i projekta kod diplomskog rada u odnosu na prethodne nivoe je objašnjiv time što su tu prisutni najpre zvučni zapisi, a potom i veći broj slika, koji zauzimaju mnogo više prostora nego izvorni (u sastavu *NetBeans* projekta), odnosno izvršni kod.

VII. Diskusija

Opisane laboratorijske vežbe sa domaćim zadatkom sprovedene su školske 2018/19 godine. U Tabeli 2 navedeni su podaci o broju studenata, prosečnoj oceni (aritmetičkoj sredini) i standardnoj devijaciji ocene na kontrolnoj laboratorijskoj vežbi, domaćem zadatku i diplomskom radu. Ukupan broj studenata upisanih na predmet *Računarska grafika*, koji su pristupili barem jednoj predispitnoj obavezi, bio je 35. Na kontrolnoj laboratorijskoj vežbi studenti su rešavali 17 zadataka od kojih je svaki nosio od 5 do 20 poena. U zbiru, svi zadaci su nosili 150 poena, pri čemu je broj poena koji nosi vežba ograničen na 100, tako da za maksimalan broj poena na vežbi nije bilo potebno realizovati sve zadatke. Laboratorijska vežba je trajala 120 min. S obzirom da je procenjena razlika u broju linija koda softvera razvijenog na kontrolnoj laboratorijskoj vežbi (kada je realizovano svih 17 zadataka) i skeleta koda koji se studentima stavlja na raspolaganje, nešto manje od 500 linija koda, za ostvarivanje maksimalnog broja poena studentima je stavljeno u zadatak da za 120 minuta za realizuju nešto više od 300 linija koda za maksimalni broj poena, što svakako zahteva visok stepen znanja i veštine programiranja 3D grafike.

TABELA II
Statistika predispitnih obaveza i diplomskih radova

	Broj studenata	Prosečna ocena	Standardna devijacija
Kontrolna lab vežba	26	53.78	30.20
Domaći zadatak	24	94.50	12.47
Diplomski rad	4	10	0

Pri analizi prikazanih rezultata treba imati u vidu činjenicu da poeni sa domaćeg zadatka koji se brani u prvom ispitnom roku, a koji predstavlja nadgradnju video-igre sa kontrolne laboratorijske vežbe, mogu da kompenziju poene sa te laboratorijske vežbe. To je jedan od razloga relativno niske prosečne ocene i velike standardne devijacije ocene na laboratorijskoj vežbi. Drugi razlog relativno male prosečne ocene je ograničeno vreme vežbe u kojem je potrebno realizovati veći broj netrivijalnih zadataka vezanih za funkcionalnosti i grafički dizajn igre.

S obzirom na činjenicu da se poeni sa laboratorijske vežbe mogu kompenzovati poenima osvojenim kroz domaći zadatak, kao i činjenicu da nije potrebno realizovati sve zadatke za maksimalni broj poena, već student može da izabere podskup zadataka koje će realizovati, nije od presudnog značaja da se broj i težina zadataka precizno prilagode raspoloživom vremenu za njihovo rešavanje. Činjenica da su dva od 26 studenata osvojila 99% i 100% poena na laboratorijskoj vežbi, pokazuje da je raspoloživo vreme bilo dovoljno za realizaciju potrebnog broja zadataka i osvajanje maksimalnog broja poena. Ključnu ulogu kontrolna laboratorijska vežba odigrala je kroz stimulisanje oko dve trećine studenata da blagovremeno počnu sa proučavanjem gradiva iz 3D grafike, provere svoje znanje i veštinu programiranja kroz izradu laboratorijske vežbe, kako bi bolje pripremljeni, ubrzo posle sprovedene vežbe, započeli rad na relativno kompleksnom domaćem zadatku i odbranili ga u prvom ispitnom roku. Izuzetno visoka prosečna ocena i relativno mala standardna devijacija ocene domaćih zadataka 24 studenta, od 26 koji su radili kontrolnu laboratorijsku vežbu, ide u prilog navedenoj pretpostavci o ključnoj ulozi laboratorijske vežbe u motivaciji učenika.

VIII. Zaključak

Iskustvo izrade ovog rada nesumnjivo ide u prilog konstataciji da je razvoj 3D video-igre zahtevan posao. Potrebno je uložiti značajno vreme i trud da bi se dizajnirala scena, kreirao scenario igre, a zatim osmislio i napisao lepo organizovan kod, koji je održiv i proširiv. U konkretnom slučaju evolutivnog razvoja igre za potrebe najpre laboratorijskih vežbi, a zatim domaćeg zadatka (projekta), da bi se na kraju nadogradila kroz diplomski rad, još više je naglašena potreba za održivošću i proširivošću koda. Pored toga, potrebno je imati dobru meru za kompleksnost ovih koraka u razvoju, jer je predviđeno vreme za izradu pojedinih koraka u rešavanju ograničeno.

Ograničenje koje je prisutno za postizanje spektakularnijih rezultata jeste sama tehnologija oslonjena na biblioteku *JavaFX*. Iako veoma praktična za akademske potrebe, ova biblioteka nema mogućnosti za lako postizanje vizuelno atraktivnih rezultata. Međutim, programerski izazovi ne zavise od toga. Kako su laboratorijske vežbe i domaći zadatak iz *Računarske grafike* praktični programerski izazovi, nivo podrške koju programer dobija kroz biblioteku *JavaFX* dobro je primeren nastavnim potrebama predmeta.

Prema iskustvu prvog autora, koji je razvio 3D video-igru *Arena* kroz svoj diplomski rad, taj razvoj se pokazao izazovnim, zahtevajući zabavnu dozu kreativnosti u rešavanju problema. Aspekti dizajniranja scene i scenarija igre uz slobodu u osmišljavanju iskustva koje igra pruža igraču, dali su izuzetnu motivaciju za razvoj pokazne i kontrolne laboratorijske vežbe, domaćeg zadatka na predmetu *Računarska grafika* i, na kraju, softvera ciljne igre realizovanog kroz diplomski rad.

Iako je realizovana igra kroz ovaj rad potpuno funkcionalna, ona je samo praktičan rezultat diplomskog rada i može se dalje nadograđivati po mnogim aspektima, sa ciljem da dostigne nivo master rada ili čak produkcioni nivo. Najpre, dizajn igre je relativno skroman i mogao bi se značajno unaprediti korišćenjem realističnijih 3D modela objekata i maštovitijih tekstura. Zatim bi se, po ugledu na mnoge akcione igre, mogli uključiti i protivnici u prostorijama, koji vođeni veštačkom inteligencijom beže od igrača i napadaju ga, pucajući na njega, dok im igrač uzvraća vatrom iz različitih oružja. Pored ovakvih dodatnih funkcionalnosti, igra bi se mogla proširiti kvizom znanja, tako što igrač po kompletiranju jednog nivoa (po završenom boravku u jednoj prostoriji) mora da reši kratak kviz znanja da bi prešao na sledeći nivo, odnosno ušao u sledeću prostoriju. I u toku boravka u jednoj prostoriji, mogli bi se pojavljivati novi predmeti koji nose veći broj poena igraču kada ih sakupi i odgovori na pitanje koje mu se tom prilikom postavi. Na taj način bi zabavna igra, realizovana u ovom radu, prerasla u obrazovnu igru upotrebljivu na različitim nivoima obrazovanja i bez ograničenja u oblasti obrazovanja.

DODATAK

Kompletno uputstvo za korišćenje sa detaljnim opisom funkcionalnosti igre (Prilog A), celovit UML model igre (Prilog B) i izvršna verzija igre *Arena*, mogu se preuzeti sa adrese:
https://www.dropbox.com/sh/4ggphvj83owcfvg/AAArlFNMO8mt74quPi4fxZgra?dl=0

Postavke zadataka za laboratorijske vežbe, domaćeg zadatka (projekta) sa dopunskim zahtevima za diplomski i master rad, kao i izvršne verzije skeleta i implementiranih programa sa laboratorijskih vežbi, mogu se preuzeti sa adrese:
https://www.dropbox.com/sh/ooso0olzky7jy65/AADrAqLJ5r_n5f_LBRCyDI5Xa?dl=0

ZAHVALNICA

Igor Tartalja je angažovan na projektima TR32039 i TR32047 finansiranim od strane Ministarstva prosvete, nauke i tehnološkog razvoja.

LITERATURA

[1] Hughes, J. F., Van Dam, A., McGuire, M., Sklar, D. F., Foley, J. D., Feiner, S. K., Akeley, K., Computer Graphics: Principles and Practice (Third Edition), Addison-Wesley, 1995.
[2] Sharan, K., Learn JavaFX 8: Building User Experience and Interfaces with Java 8, Apress, 2015.
[3] NetBeans razvojno okruženje, zvanična veb stranica: https://netbeans.org/, pristupano – jul 2020.
[4] Video game, Wikipedia, veb stranica: https://en.wikipedia.org/wiki/Video_game, pristupano – jul 2020.
[5] Action game,Wikipedia, veb stranica: https://en.wikipedia.org/wiki/Action_game, pristupano – jul 2020.
[6] Platform game, Wikipedia, veb stranica: https://en.wikipedia.org/wiki/Platform_game, pristupano – jul 2020.
[7] First-person (video games), Wikipedia, veb stranica: https://en.wikipedia.org/wiki/First-person_(video_games), pristupano – jul 2020.
[8] Rayman 2: The Great Escape: https://store.ubi.com/ie/game?pid=56c4947e88a7e300458b465c&dwvar_56c4947e88a7e300458b465c_Platform=pcdl&edition=Standard%20Edition&source=detail, pristupano – septembar 2020.
[9] Croc 2, Wikipedia, veb stranica: https://en.wikipedia.org/wiki/Croc_2, pristupano – jul 2020.
[10] Half-Life: https://www.half-life.com/en/halflife, pristupano – septembar 2020.
[11] Package java.util.stream, zvanična dokumentacija: https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html, pristupano – jul 2020.
[12] SourceMonitor, alat za metriku koda, zvanična veb-stranica: http://www.campwoodsw.com/sourcemonitor.html, pristupano – jul 2020.

ABSTRACT

The paper describes an experience of incremental development of a 3D video-game through its evolution from the code skeleton presented to students at a tutorial laboratory exercise in the Computer Graphics course, through a control laboratory exercise and later homework (project), all the way to the student 's diploma thesis. The goal of the developed game is to collect as many points as possible in a limited time and survive in an arena with several rooms, avoiding obstacles, while collecting various bonuses. The software is modeled in the standard UML language (*Unified Modeling Language*), and implemented in the *Java* programming language, using the *JavaFX* library.

Incremental development of the 3D video-game *Arena* through students' practical work

Vojislav Bogosavljević, Igor Tartalja

Jedna realizacija signalizacije niskog i servisnog nivoa u programskoj podršci ATSC 3.0 srednjeg sloja

Lazar Švonja, Mladen Ilić, Ilija Bašičević

***Apstrakt*—U ovom radu su opisane osnovne karakteristike i principi ATSC (eng. Advanced Television System Comitee) standarda, kao i implementacija programske podrške za ATSC 3.0. ATSC 3.0 je dominantan TV standard u Severnoj Americi. Poseban akcenat biće na signalizaciji niskog i servisnog nivoa u okviru modula za upravljanje servisima. Ideja ovog rada je pored ostalog da ukaže na specifičnosti ATSC standarda, gde se kao glavni problem javlja rukovanje ROUTE i MMTP paketima. Dato je jedno rešenje obrade signalnih paketa, koje kao rezultat daje podatke u obliku pogodnom za dalju obradu i reprodukciju. Rešenje je verifikovano korišćenjem namenske Java aplikacije za Android operativni sistem.**

***Ključne reči*—multimedija; digitalna televizija; digitalni TV prijemnik; ATSC;**

I. Uvod

ATSC (eng. Advanced Television Systems Committee) predstavlja standard dopremanja digitalne televizije, usvojen od strane nekoliko zemalja, uključujući i Severnu Ameriku, prilikom prelaska sa analogne na digitalnu televiziju.

ATSC 3.0 se zasniva na IP (eng. Internet Protocol) protokolu, iako to ne znači da je za gledanje televizije potreban internet.

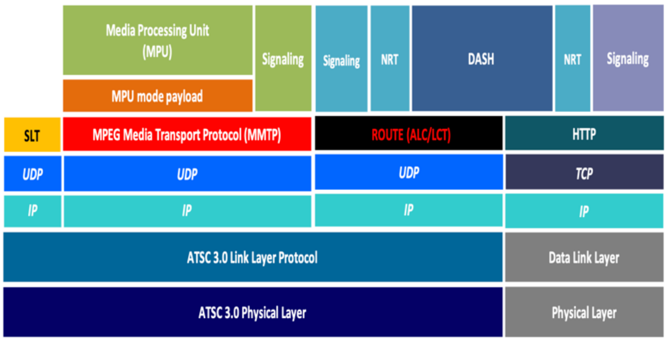
ATSC 3.0 sadržaj se isporučije krajnjim korisnicima putem tri funkcionalna sloja (Sl. 1.):

- Fizički sloj (eng. Physical layer)
- Sloj za isporuku sadržaja (eng. Delivery layer)
- Sloj za upravljanje servisima (eng. Service Management layer)

Fizički sloj obezbeđuje mehanizam putem kog se signali, najava servisa i IP paketi prenose emisionim (eng. Broadcast) ili širokopojasnim (eng. Broadband) putem.

Sloj za isporuku sadržaja obezbeđuje objekat, kao i funkcionalnost transporta objekata. Zasniva se na MMTP (eng. MPEG Media Transport Protocol) protokolu ili na ROUTE (eng. Real-Time Object Delivery over Unidirectional Transport) protokolu, koji rade na UDP/IP multicast-u koji se isporučuje emisionim putem, kao i na HTTP protokolu koji radi na TCP/IP unicast-u i isporučuje se širokopojasnim putem.

Sloj za upravljanje servisima primarno podržava sredstva za otkrivanje i akviziciju za omogućavanje različitih vrsta servisa digitalne televizije.



Sl. 1. Programska podrška ATSC 3.0 prijemnika [1]

Signalizacija servisa pruža informacije o otkrivanju i opisu servisa, i sadrži dve funkcionalne komponente:

- Signalizacija niskog nivoa (eng. Low Level Signaling - LLS)
- Signalizacija servisnog nivoa (eng. Service Layer Signaling - SLS)

Navedene komponente predstavljaju informacije neophodne za otkrivanje i korišćenje ATSC 3.0 servisa, a predmet ovog rada biće jedno rešenje signalizacije niskog i servisnog nivoa u srednjem sloju digitalnog televizijskog ATSC 3.0 prijemnika.

II. Teorijske osnove

A. *ROUTE i MMTP protokol*

Za sadržaj isporučen emisionim putem, u slučaju ROUTE sadržaja, SLS se prenosi u jednom od transportnih kanala koji sadrže ROUTE sesiju, dok se u slučaju MMTP sadržaja SLS prenosi MMTP signalnim porukama, u oba slučaja sa odgovarajućim periodom ponavljanja kako bi se omogućilo brzo pridruživanje i prebacivanje kanala. Za sadržaj isporučen širokopojasnim putem, SLS se prenosi preko TCP/IP protokola.

ROUTE sesija sadrži jedan ili više LCT (eng. Layered Coding Transport) kanala, koji u sebi nose neophodan sadržaj za ATSC 3.0 servis, kao što je audio, video ili prevod. Medijski sadržaj je prikazan u vidu ISO BMFF (eng. ISO Base Media File Format) segmenata.

MMTP sesija sadrži jedan ili više MMTP paketa, koji prenose MMT signalne poruke ili komponente u obliku MPU-

Lazar Švonja, Nacionalni institut za nauku i razvoj RT-RK, Narodnog fronta 23A, 21000 Novi Sad, Srbija (e-mail: lazar.svonja@rt-rk.com).

Mladen Ilić, Nacionalni institut za nauku i razvoj RT-RK, Narodnog fronta 23A, 21000 Novi Sad, Srbija (e-mail: mladen.ilic@rt-rk.com).

Ilija Bašičević, Fakultet tehnickih nauka, Trg Dositeja Obradovića 6, 21000 Novi Sad, Srbija (e-mail: ilibas@uns.ac.rs).

a (eng. Media Processing Unit) za MMT.

Za isporuku servisa baziranih na aplikaciji ili metapodataka sistema, kao što su informacije o signalizaciji servisa i aplikacija, LCT kanal sadrži stavke sadržaja bazirane na datotekama.

B. Identifikatori servisa

Svaki servis se identifikuje pomoću dva oblika identifikatora servisa:

- Kompaktnog oblika koji se koristi u SLT tabeli i jedinstven je samo u području emitovanja
- Globalno jedinstvenog oblika koji se koristi u ESG-u

ROUTE sesija identifikuje se IP adresom izvora, IP adresom odredišta, i brojem odredišnog izlaza. Za identifikaciju LCT kanala služi identifikator transportne sesije (eng. Transport Session Identifier - TSI), koji je jedinstven u okviru roditeljske ROUTE sesije kao i u okviru svih ROUTE sesija definisanih za dati servis u S-TSID-u (eng. Service-based Transport Session Instance Description).

S-TSID predstavlja signalnu strukturu u kojoj su prikazana svojstva zajednička za LCT kanale, kao i određena svojstva jedinstvena za pojedine LCT kanale i deo je signalizacije niskog nivoa. Svaki LCT kanal prenosi se preko jednog PLP-a (eng. Physical Layer Pipe), koji predstavlja deo radio frekventnog kanala koji ima jasno definisanu modulaciju i uslove kodovanja.

MMTP sesija identifikuje se odredišnom IP adresom i brojem odredišnog izlaza. MMTP protok paketa identifikuje se identifikatorom paketa (eng. packet_id) koji je jedinstven u okviru roditeljske MMTP sesije. Svojstva zajednička za sve MMTP pakete i određena svojstva pojedinačnih paketa predstavljaju deo signalizacije servisnog nivoa. Svojstva za svaku MMTP sesiju dobijaju se od strane MMT signalnih poruka, koje se mogu prenositi u okviru povezanih MMTP sesija.

C. Signalizacija niskog nivoa

Singnalizacija niskog nivoa predstavlja signalne informacije koje se nalaze u korisnom sadržaju IP paketa, pod određenom adresom i portom, a zadužene su za brzu pretragu kanala i dobavljanje servisa od strane primaoca. Tipovi LLS informacija su skladišteni u obliku LLS tabele i oni su:

- Tabela liste servisa (eng. Service List Table - SLT)
- Roditeljska kontrola (eng. Rating Region Table)
- Sistemsko vreme (eng. System Time fragment)
- Tabela upozorenja (eng. Advanced Emergency Alert Table)
- Fragment za prikaz poruka (eng. Onscreen Message Notification fragment)

D. Signalizacija servisnog nivoa

Signalizacija servisnog nivoa predstavlja signalizaciju koja obezbeđuje informacije za otkrivanje i dobavljanje ATSC 3.0 servisa i njegovih komponenti.

Za isporuku servisa putem ROUTE protokola, SLS tabela za svaki servis opisuje karekteristike servisa, sposobnosti prijemnika potrebne za smislenu prezentaciju servisa, kao i dostupnost i pravila u vezi pristupa servisima za popravljanje datoteka pomoću prijemnika.

U slučaju DASH (eng. Dynamic Adaptive Streaming over HTTP) servisa, SLS uključuje i opis skupa korisničkog servisa (USBD), S-TSID, opis prezentacije DASH medija (MPD), a može uključivati i HTTP stranice za opis lokacije (HELD), kao i tablicu dostupnosti regionalnih servisa (RSAT).

Za servise isporučene putem MMTP protokola, SLS tabela za svaki servis, pored osnovnih informacija o karakteristikama servisa, uključuje i USBD fragment, tablicu MMT paketa (MP), a može sadržati i HELD.

Signalizacija servisnog nivoa usredsređena je na osnovne atribute servisa, a posebno na one atribute potrebne za dobavljanje servisa. Podaci o servisima namenjeni gledaocima nazivaju se ESG podaci.

III. Opis problema i koncept rešenja

A. Arhitektura ATSC 3.0 DTV (Digital television) srednjeg sloja

ATSC 3.0 DTV srednji sloj predstavlja programsku podršku koja treba da omogući:

- Rukovanje ATSC 3.0 instalacijom i listama servisa
- Reprodukciju servisa uživo
- Elektronski vodič za servise
- Sistem za upozorenje
- Prevod
- Prateće uređaje
- Roditeljsku kontrolu
- Interaktivni sadržaj

Za potrebe ovog rešenja, ATSC 3.0 je nezavisna komponenta, pisana u programskom jeziku C, koja se može koristiti na različitim okruženjima, kao što su Android TV ili AOSP (eng. Android Open Source project) okruženje, Linux ili na bilo kom drugom okruženju koje se oslanja na SDK (eng. Software Development Kit).

Programsku podršku ATSC 3.0 DTV srednjeg sloja čine sledeći slojevi:

- Sloj za apstrakciju DTV srednjeg sloja koji omogućava pozive ka DTV srednjem sloju iz aplikativnog dela programske podrške
- ATSC srednji sloj – ključna komponenta zadužena za obradu signalizacije niskog i servisnog nivoa, upravljanje listama servisa…
- Sloj za apstrakciju fizičke arhitekture koji pruža podršku za rukovanje fizičkim komponentama
- Media Player – komponenta zadužena za reprodukciju ATSC 3.0 sadržaja

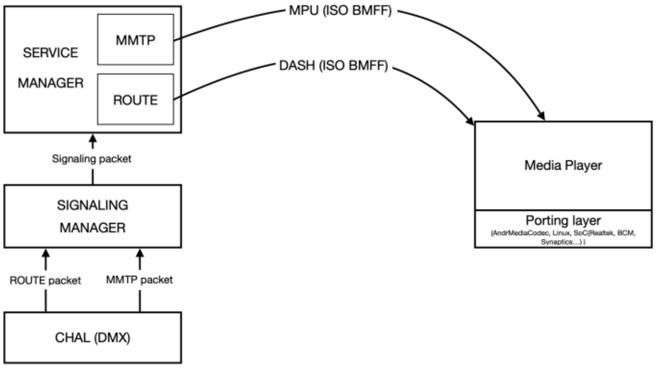
Ključni moduli ATSC srednjeg sloja implementiraju i procesuiraju signalizaciju niskog i servisnog nivoa:

- Modul za upravljanje signalima (eng. Signaling Manager)
- Modul za upravljanje servisima (eng. Service Manager)

Modul za upravljanje signalima se koristi za rukovanje sirovim podacima, od kojih napravi signalne objekte predstavljene u vidu strukture podataka. Dobijeni objekti se dalje prosleđuju na obradu modulu za upravljanje servisima, koji ih sklapa u konkretne objekte i šalje na izlaz korisniku preko komponente Media Player.

B. *Opis problema*

Kada je reč o implementaciji ATSC 3.0 srednjeg sloja, glavni problem predstavljala je obrada MMTP i ROUTE sadržaja u cilju stvaranja podataka koje Media Player može da reprodukuje. Modul za upravljanje signalima, kao i modul za upravljanje servisima su imali ključnu ulogu u obradi podataka i njihovoj pripremi za Media Player. Ideja je da se deo podataka koji pristiže iz sloja za apstrakciju fizičke arhitekture prosleđuje modulu za upravljanje signalima, dok se ROUTE i MMTP paketi koji takođe pristižu iz sloja za apstrakciju fizičke arhitekture, prosleđuju do modula za upravljanje servisima. U modulu za upravljanje servisima se zatim izvršava prihvatanje paketa, njihova obrada, i na kraju slanje sadržaja u Media Player, u vidu MPU ili DASH segmenata, koji predstavljaju oblike prepoznatljivog ISO BMFF formata, što je prikazano na Sl.2.



Sl. 2. Tok podataka u ATSC 3.0 srednjem sloju

U daljem tekstu biće detaljno opisana implementacija modula za upravljanje signalima, kao i modula za upravljanje servisima.

C. *Koncept rešenja*

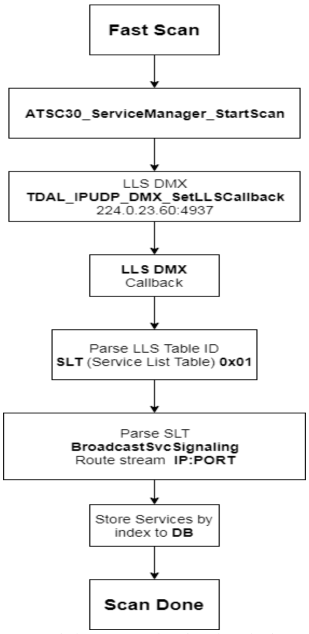
Realizacija signalizacije niskog i servisnog nivoa biće objašena na primerima implementacije modula za skreniranje kanala, kao i za prebacivanje kanala.

Kada je modul za skeniranje kanala u pitanju, početnu tačku procesa predstavljala je pretraga standardom definisane IP adrese i izlaza. Standardnom definisana IP adresa na kojoj se nalaze LLS paketi je 224.0.23.60 i izlaz 4937/udp. Kreira se filter na nivou demultipleksera, gde se kao kriterijum za filtriranje upotrebljava lokacija koja je unapred definisana.

Prilikom pronalaženja paketa, demultiplekser obaveštava ATSC srednji sloj i šalje LLS objekte na obradu. Prilikom parsiranja tabele zaglavlja LLS paketa pronalazi se verzija i tip tabele. Na osnovu tipa tabele zaključuje se koji parser će se koristiti za procesiranje, a na osnovu verzije tabele dalje se radi ažuriranje obrađenih verzija, kako se ne bi ponavljala obrada. Sledeći korak je parsiranje parametra LLS objekta TABLE ID, iz kojeg se dobija SLT tabela.

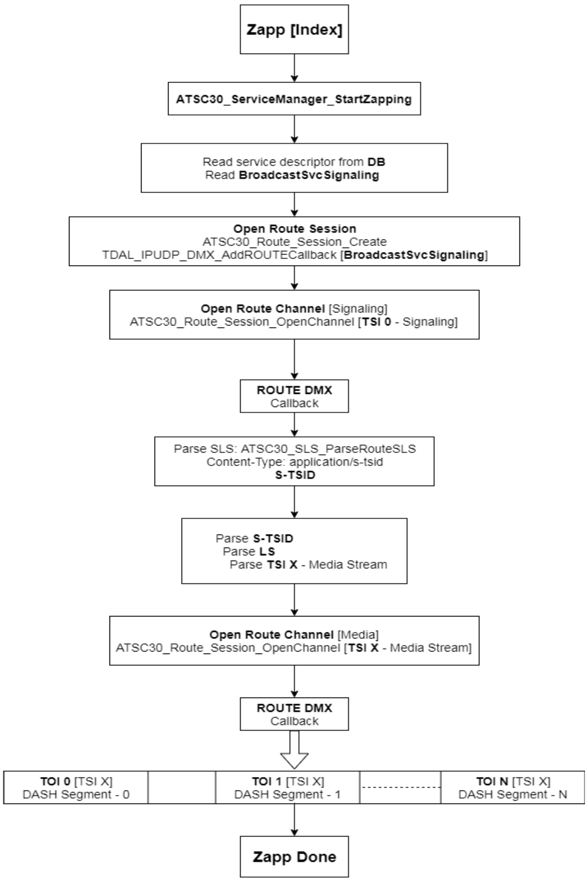
Daljim parsiranjem SLT tabele dobijamo karakteristiku BroadcastSvcSignaling, koja je sa svojim elementima zadužena za pružanje širokopojasnih signalnih informacija. Iz tog razloga možemo da pristupimo tipu protokola, IP adresi, kao i izlazu individualnog kanala.

Na samom kraju obrade objekta SLT tabele isparsirani podaci se smeštaju u bazu podataka.



Sl. 3. Tok izvršavanja skeniranja kanala

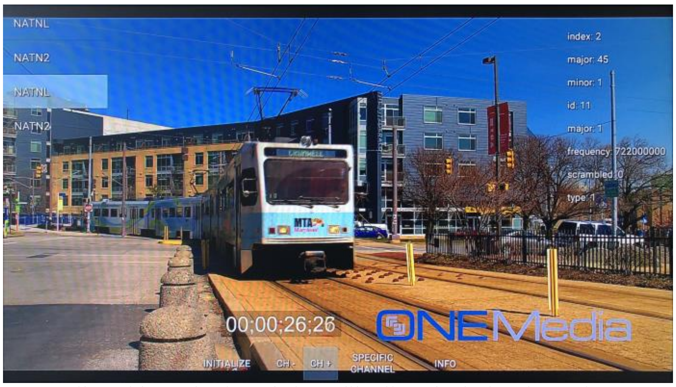
Prebacivanje kanala počinje pozivom funkcije za pokretanje kanala, koja se nalazi u ATSC srednjem sloju. Potom se upotrebljavaju prethodno isparsirana polja karakteristike BroadcastSvcSignaling iz baze podataka. Zatim demultiplekser otvara sesiju, koja omogućava filtriranje druge sesije po IP adresi i izlazu, uz pomoć karakteristike BroadcastSvcSignaling. U okviru druge sesije se otvara kanal rezervisan za signalne podatke i ima uniformnu oznaku TSI 0. Kanal predstavlja još jedan vid apstrakcije iznad sesije. Zatim se otvaraju i ostali kanali u okviru iste sesije, koju dalje obrađuje ATSC srednji sloj. Modul za upravljanje signalima vrši parsiranje zaglavlja LCT objekta, koji pripada SLS paketu, i kao rezultat daje S-TSID objekat. S-TSID objekat nam daje TSI broj veći od 0, koji je zadužen za razvrstavanje kanala po tipu sadržaja (audio, video itd.). Na kraju dobijamo i TOI (eng. Transport Object Identifier) objekte, koji se nalaze u okviru TSI kanala. Svaki TOI objekat je sačinjen od nekoliko IP paketa. Jedan TOI objekat predstavlja jedan audio ili video segment, koji sadrži npr. jednu sekundu audio ili video sadržaja.


Zapp [Index]
ATSC30_ServiceManager_StartZapping
Read service descriptor from DB
Read BroadcastSvcSignaling
Open Route Session
ATSC30_Route_Session_Create
TDAL_IPUDP_DMX_AddROUTECallback [BroadcastSvcSignaling]
Open Route Channel [Signaling]
ATSC30_Route_Session_OpenChannel [TSI 0 - Signaling]
ROUTE DMX
Callback
Parse SLS: ATSC30_SLS_ParseRouteSLS
Content-Type: application/s-tsid
S-TSID
Parse S-TSID
Parse LS
Parse TSI X - Media Stream
Open Route Channel [Media]
ATSC30_Route_Session_OpenChannel [TSI X - Media Stream]
ROUTE DMX
Callback
TOI 0 [TSI X]
DASH Segment - 0
TOI 1 [TSI X]
DASH Segment - 1
TOI N [TSI X]
DASH Segment - N
Zapp Done

Sl. 4. Tok izvršavanja prebacivanja kanala

IV. Verifikacija i testiranje

Za verifikovanje i testiranje ovog rešenja korišćene su dve različite aplikacije, od kojih je jedna konzolna aplikacija, dok je druga aplikacija sa korisnim okruženjem. Konzolna aplikacija je pokretana na Linux i Android operativnim sistemima i korišćena je za testiranje funkcija koje pristupaju signalnim podacima, verifikaciju skeniranja kanala, menjanja kanala, ali i za potvrdu stabilnosti datih modula. Nedostatak verifikacije i testiranja putem konzolne aplikacije je u tome što konzolna aplikacija nema mogućnost prikaza audio i video signala. Iz tog razloga je kreirana i aplikacija sa korisnim okruženjem, koja je napisana u Java programskom jeziku i koristi se na DTV uređaju koji pokreće Android operativni sistem. Koristi se i za prikazivanje audio i video signala, pored svih funkcionalnosti koje poseduje i konzolna aplikacija. Prikaz testiranja u okviru aplikacije sa korisnim okruženjem se može videti na Sl. 5.



Sl. 5. Prikaz video sadržaja u okviru aplikacije sa korisnim okruženjem

U svrhu testiranja stabilnosti rešenja napravljene su testne skripte koje pokreću konzolnu aplikaciju i testiraju sledeće slučajeve:

- 100-1000 promena kanala sa različitim vremenskim intervalom na svake 2 sekunde, 10 sekundi i 60 sekundi
- 100 sekvenci skeniranja kanala; pri brzom i kompletnom skeniranju kanala
- Testiranje sekvenci skeniranja i promene kanala u proizvoljnom vremenskom intervalu i broju ponavljanja
- 24 sata reprodukcije audio i video sadržaja

V. Zaključak

U ovom radu predstavljeno je jedno rešenje signalizacije niskog i servisnog nivoa ATSC 3.0 standarda u digitalnim TV uređajima. Implementacija ovog rešenja u kojem se usled programskog nivoa koji apstrahuje platformu, omogućava potpuna nezavisnost od fizičke platforme i operativnog sistema, što kao rezultat nudi korišćenje ovog rešenja na raznim platformama. Dalja proširenja idu u pravcu integracije u srednji sloj DTV uređaja, razvijanja i unapređivanja postojećih funkcionalnosti ATSC 3.0 srednjeg sloja, kao i dodavanje elektronskog vodiča za servise, prevode i roditeljsku kontrolu.

Zahvalnica

Ovo istraživanje (ovaj rad) je podržan(o) od stane Ministarstva prosvete, nauke i tehnološkog razvoja kroz projekat broj 451-03-68/2020-14/200156: "Inovativna naučna i umetnička ispitivanja iz domena delatnosti FTN-a".

Literatura

[1] ATSC Standard: "Signaling, Delivery, Synchronization, and Error Protection (A/331)," 20.06.2019.

[2] W. Zia, T. Stockhammer, K. Walker, "Demonstrating ATSC ROUTE-DASH Delivery," MMSys'16, Klagenfurt, Austria, 10-13.05.2016.

[3] DASH Industry Forum, Guidelines for Implementation: "DASH-IF Interoperability Point for ATSC 3.0," 12.06.2018.

[4] IETF: RFC 7231, "Hypertext Transfer Protocol -- HTTP/1.1," Internet Engineering Task Force, Reston, VA, June 2014. http://tools.ietf.org/html/rfc7231

Abstract

This paper presents most important characteristics and principles of ATSC (Advanced Television System Comitee) standard and an implementation of ATSC 3.0 software. ATSC 3.0 is a dominant TV standard in North America. Focus in this paper is on Low Level and Service Layer Signalization, in the context of Service Manager module. The goal of this paper is also to highlight the specific properties of the ATSC standard, in which the core problem is handling of ROUTE and MMTP packets. A realization of the signal packet processing is provided, which on the output delivers data in a format that is suitable for further processing and playback. The system has been verified using a custom tailored Java application for the Android operating system.

A realization of Low Level and Service Layer Signaling in ATSC 3.0 middleware

Lazar Švonja, Mladen Ilić, Ilija Bašičević

Primena dvonivovskog algoritma za detekciju ivica modelovanog u Nadgrađenoj Petri-mreži

Perica Štrbac, Vukman Korać, Miloš Pejanović

***Apstrakt*—U radu je predložen dvonivovski algoritam za detekciju ivica. Prvi nivo obuhvata: transformaciju slike u boji u sivu sliku, razvlačenje kontrasta, primenu Gausovog zamućenja i primenu dvostrukog praga. Drugi nivo obuhvata delove Kanijevog algoritma: računanje gradijenta, potiskivanje nemaksimuma te zadržavanje piksela jakih ivica i piksela slabih ivica koje su povezane sa jakim ivicama. Model predloženog algoritma je modelovan u Nadgrađenim Petri-mrežama (UPN - eng. Upgraded Petri-nets), a onda je model transformisan u konkretan računarski program. Prikazani su eksperimentalni rezultati primene predloženog algoritma nad odabranim skupom slika i poređenja sa Kanijevim algortimom.**

***Ključne reči*—detekcija ivica; razvlačenje kontrasta, Kani, Nadgrađene Petri-mreže.**

I. Uvod

Izdvajanje ivica iz slike, u cilju vršenja računarske interpretacije i dalje analize slike, ima značajnu primenu u različitim oblastima, kao što su analiza satelitskih snimaka, analiza medicinskih snimaka, identifikacija u sistemima za razvrstavanje pošte, navođenje projektila u vojnim primenama, itd [1].

U ovom radu je urađena eksperimentalna primena dvonovovskog algoritma nad skupom odabranih satelitskih slika koje su javno dostupne sa različitih satelita. Cilj eksperimenta je da se varijacijom ulaznih parametara po fazama dvonivovskog algoritma dobije optimalna slika ivica iz slike.

Algoritam smo izveli u dva nivoa. Prvi nivo obuhvata pripremu slike za lakše određivanje piksela koji pripadaju ivicama i drugi koji selektuje piksele koji pripadaju jakim ili su povezani sa jakim ivicama.

Za modelovanje, simulaciju i analizu koristili smo Nadgrađene Petri-mreže, a za eksperimente softverski program koji je dobijen transformacijom pripadnog UPN modela [2].

Skup slika koje smo koristili u eksperimentima su javno dostupne slike snimeljene sa satelita: Kanopus V1, Landsat 8, UK-DMC2, Aster, LANDSAT 7 ETM+, Ikonos i QuickBird.

II. Dvonivovski algoritam za detekciju ivica

Predloženi dvonivovski algoritam za detekciju ivica uključuje kao što sledi:

- prvi nivo obuhvata učitavanje slike u boji sa tri kolorne ravni i njenu transformaciju u sivu sliku sa jednom (sivom) ravni, primenu razvlačenja konstrasta (eng. constrast stretching) sive slike, delovanje Gausovog zamućenja (eng. Gaussian blur) nad novodobijenom slikom te primenu dvostrukog praga (eng. threshold) nad zamućenom slikom;
- drugi nivo obuhvata računanje gradijenta prema pikselima slike koja je rezultat prvog nivoa, potiskivanje nemaksimuma, zadržavanje piksela jakih ivica te zadržavanje piksela slabih ivica koje su povezane sa jakom ivicom.

Drugi nivo pripada delovima Kanijevog (eng. Canny) algoritma za detekciju ivica [3].

A. *Transformacija slike u boji u sivu sliku*

Transformacija slike u boji u sivu sliku urađeno je korišćenjem sledeće formule:

$$P_g(x,y) = k_r \cdot P_r(x,y) + k_g \cdot P_g(x,y) + k_b \cdot P_b(x,y) \qquad (1)$$

gde su:

$P_g(x,y)$ – vrednost piksela sive slike na poziciji x,y;

$P_r(x,y)$, $P_g(x,y)$, $P_b(x,y)$ – R, G i B vrednosti piksela slike u boji na poziciji x,y, respektivno;

k_r, k_g, k_b – koeficijenti učešća R, G i B vrednosti piksela slike u boji u izračunavanju pripadnog piksela sive slike, pri čemu $k_r+k_g+k_b$ =1.0.

Za kreiranje sive slike može se koristiti aritmetička sredina R, G i B vrednosti datog piksela što znači da su vrednosti koeficijenata $k_r = k_g = k_b =$ 1/3. Za dobijanje sive slike koja je bolja za percepciju ljudskim okom koriste se vrednosti k_r=0.299, k_b=0.587 te k_b=0.114, čime se forsira učešće zelene komponente piksela slike u boji u računanju pripadnog sivog piksela [4].

B. *Razvlačenje konstrasta sive slike*

Razvlačenje konstrasta slike urađeno je korišćenjem sledeće formule:

$$P_o(x,y) = (P_i(x,y) - min_i) \cdot (\frac{max_o - min_o}{max_i - min_i} + min_o) \qquad (2)$$

gde su:

$P_o(x,y)$- vrednost izlaznog piksela na koordinatama x,y;

$P_i(x,y)$- vrednost ulaznog piksela na koordinatama x,y;

min_i - minimalna vrednost piksela u ulaznoj slici;

max_i - maksimalna vrednost piksela u ulaznoj slici;

min_o - minimalna vrednost piksela u izlaznoj slici;

max_o - maksimalna vrednost piksela u izlaznoj slici.

Perica Štrbac – Odsek Visoka škola elektrotehnike i računarstva, Akademija tehničko-umetničkih strukovnih studija u Beogradu, Vojvode Stepe 283, 11000 Beograd, Srbija (e-mail: pericas@ viser.edu.rs).

Vukman Korać – Odsek Visoka škola elektrotehnike i računarstva, Akademija tehničko-umetničkih strukovnih studija u Beogradu, Vojvode Stepe 283, 11000 Beograd, Srbija (e-mail: vukmank@ viser.edu.rs).

Pejanović Miloš – Odsek Visoka škola elektrotehnike i računarstva, Akademija tehničko-umetničkih strukovnih studija u Beogradu, Vojvode Stepe 283, 11000 Beograd, Srbija (e-mail: pejanovicm@ viser.edu.rs).

Korišćenjem razvlačenja konstrasta od slike koja je zamućenija npr. slika u magli dobiće se slika koja ima više razlučivih detalja [5].

C. *Gausovo zamućenje*

Gausovo zamućenje je urađeno korišćenjem sledeće formule:

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3}$$

gde su:

G(x,y) – vrednost Gausovog kernela u poziciji x,y

x, y – koordinate ćelije u kernelu

σ - disperzija

e – Ojlerov broj (≈ 2,718)

π - Ludolfov broj (≈ 3,141)

Gausovo zamućenje je nelinearno i daje kao rezultat glatko zamućenje [6].

D. *Dvostruki prag*

Dvostruki prag je realizovan prema formuli:

$$P_g(x,y)=\begin{cases} 0, P_g(x,y)\leq th\_low \\ 255, P_g(x,y)\geq th\_high \\ P_g(x,y)\ , th\_low < P_g(x,y) < th\_high \end{cases} \tag{4}$$

gde su:

$P_g(x,y)$ – vrednost intenziteta piksela sive slike na koordinatama x, y.

x,y – koordinate piksela sive slike

th_low – vrednost donjeg praga

th_high – vrednost gornjeg praga

Dvostruki prag je podesan za izdvajanje više objekata na slici u odnosu na pozadinu.

E. *Računanje gradijenta*

Računanje gradijenta obuhvata računanje intenziteta gradijenta kao i smera gradijenta što je dato sledećim formulama [7]:

$$|\nabla I| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2+\left(\frac{\partial I}{\partial y}\right)^2} \tag{5}$$

$$\varphi = arctg\left(\frac{\frac{\partial I}{\partial y}}{\frac{\partial I}{\partial x}}\right) \tag{6}$$

gde su:

/∇I/ - intenzitet gradijenta

∂I/∂x – parcijalna derivacija po x

∂I/∂y – parcijalna derivacija po y

φ - smer gradijenta

Podaci o intenzitetu i smeru gradijenta za svaku tačku slike su potrebni za potiskivanje nemaskimuma.

F. *Supresija nemaksimuma*

Supresija nemaksimuma ostavlja gradijent za dati piksel ako je njegova vrednost veća od gradijenta oba susedna piksela gledano prema smeru gradijenta za dati piksel. Ako prethodno nije zadovoljeno intenzitet gradijenta za dati piksel postavljamo na nulu.

G. *Zadržavanje piksela jakih ivica*

Postavljaju se dva praga za prihvatanje piksela ivica: weak_th za piksele slabih ivica i strong_th za piksele jakih ivica. Ako je gradijent datog piksela veći od strong_th onda takav piksel pripada jakoj ivici i biće zadržan [3].

H. *Zadržavanje piksela slabih ivica koji su povezani sa jakom ivicom*

Ako su za piksel koji ne pripada jakoj ivici vrednosti gradijenta svakog njegovog susednog piksela veće od weak_th onda će takav piksel biti zadržan [3].

III. UPN MODEL PREDLOŽENOG ALGORITMA

UPN je uređena devetorka kao što sledi:

$$C = (P, T, F, B, \mu, \theta, TF, TFL, PAF)$$

gde su:

$P=\{p_1, p_2, ..., p_n\}, n>0$ - konačan neprazan skup mesta p_i;

$T=\{t_1, t_2, t_3,...,t_m\}, m>0$ - konačan neprazan skup prelaza t_j;

$F:T\times P\rightarrow N_0$ - ulazna funkcija;

$B:T\times P\rightarrow N_0$ - izlazna funkcija;

$\mu:P\rightarrow N_0$ - funkcija markiranja:

$\theta:T\times N_0\rightarrow [0,1]$ - vremenska funkcija;

$TF: T\rightarrow A$ - funkcija prelaza;

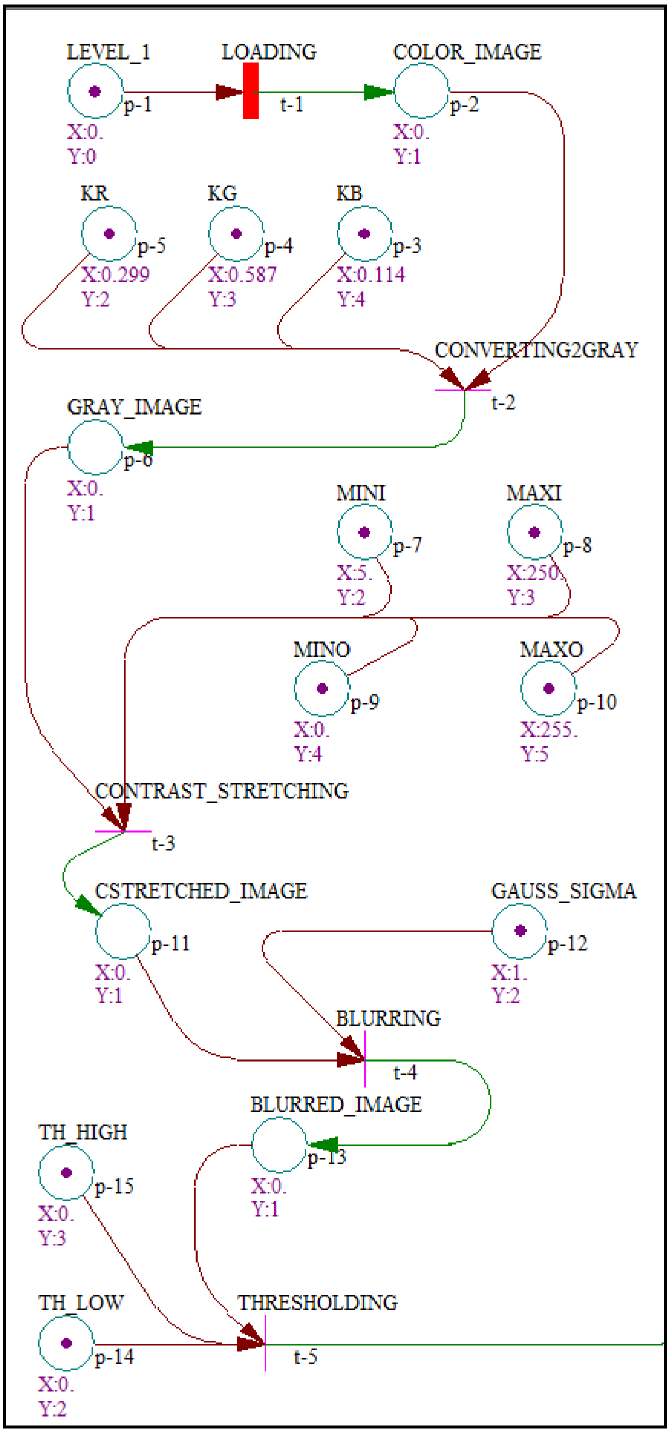
$TFL: T\rightarrow N_0$ - funkcija nivoa paljenja prelaza;

$PAF: P\rightarrow(x, y)$ - funckija atributa mesta;

Za detalje o modelovanju, simulaciji i analizi pomoću Nadgrađenih Petri-mreža videti referencu [2].

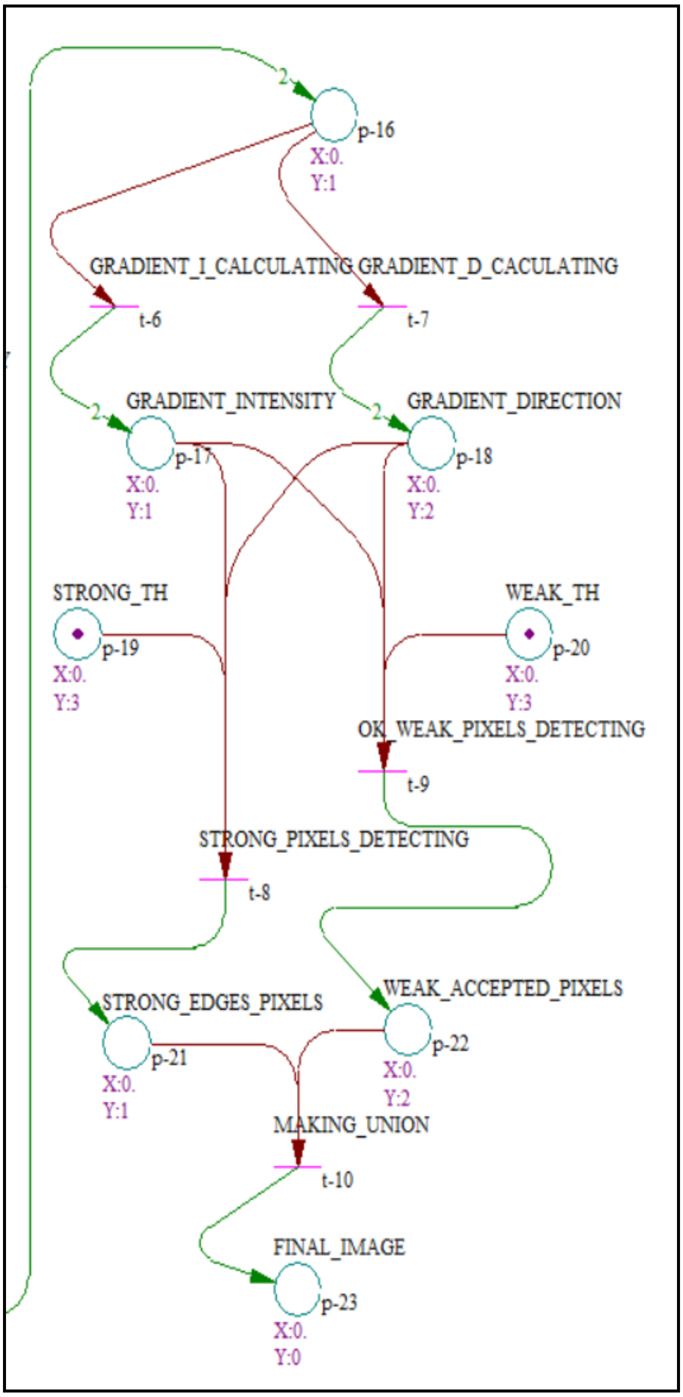
Na slikama 1 i 2 prikazano je početno označavanje UPN modela predloženog algoritma gde je μ_0=(1,0,1,1,1,0,1, 1,1,1,0,1,0,1,1,0,0,0,1,1,0,0,0).

Model na slici 1 se odnosi na prvi nivo predloženog algoritma. Mesto p-1 modeluje stanje početka prvog nivoa. Prelaz t-1 je omogućen i njegovo paljenje modeluje učitavanje slike u boji. Paljenjem prelaza t-1 menjaju se označavanja μ(p-1)=(0), μ(p-2)=1. Prelaz p-2 modeluje učitanu sliku u boji. Sada je omogućen prelaz t-2 čije paljenje modeluje transformaciju učitane slike u boji u sivu sliku korišćenjem formule (1). Ulazna mesta p-3, p-4 i p-5 modeluju koeficijente k_r, k_g i k_b, respektivno. X atributi ovih mesta se odnose na pripadne vrednosti navedenih koeficijenata, dok se Y atributi odnose na redosled operanada u operaciji dodeljenoj prelazu t-2. Paljenjem prelaza t-2 menjaju se označavanja: μ(p-2)=0, μ(p-3)= μ(p-4)= μ(p-5)=0 i μ(p-6)=1. Prelaz p-6 modeluje sivu sliku, dok mesta p-7, p-8, p-9 i p-10 modeluju argumente min_i, max_i, min_o i max_o formule (2). X atributi ovih mesta se odnose na pripadne vrednosti navedenih argumenata, dok se Y atributi odnose na redosled operanada u operaciji dodeljenoj prelazu t-3. Sada je omogućen prelaz t-3 čije paljenje modeluje razvlačenje kontrasta sive slike primenom formule (2). Paljenjem prelaza t-2 menjaju se označavanja: μ(p-6)= μ(p-7)= μ(p-8)= μ(p-9)= μ(p-10)=0 i μ(p-11)=1. Mesto p-11 modeluje rezulatat formule (2), a mesto p-12 modeluje parametar σ formule (3). Omogućen je prelaz t-4 čije paljenje modeluje konvoluciju Gausovog kernela dobijenog prema formuli (3) na sliku sa razvučenim kontrastom. Paljenjem prelaza t-4 menjaju se označavanja te je: μ(p-11)=μ(p-12)=0 i



Sl. 1. UML model prvog nivoa predloženog algoritma za detekciju ivica

μ(p-13)=1. Mesto p-13 modeluje rezultat dobijen primenom Gausovog zamućenja, dok mesta p-14 i p-15 modeluju parametre *th_low* i *th_high* formule (4). X atributi ovih mesta se odnose na pripadne vrednosti navedenih parametara, dok se Y atributi odnose na redosled operanada u operaciji dodeljenoj prelazu t-5. Sada je zapaljiv prelaz t-5 čije paljenje modeluje primenu formule (4). Paljenjem prelaza t-5 menjaju se označavanja: μ(p-13)=μ(p-14)=μ(p-15)=0 i μ(p-16)=2. Označavanje mesta p-16 ima vrednost 2 zbog dvostrukog izlaznog luka funkcije B(t-5, p-16).



Sl. 2. UML model drugog nivoa predloženog algoritma zadetekciju ivica

Ovim je završen prvi nivo predloženog algoritma.

Model na slici 2 se odnosi na drugi nivo predloženog algoritma. Mesto p-16 modeluje stanje početka drugog nivoa. Sada je omogućeno paralelno paljenje skupa prelaza {t-6, t-7} zbog μ(p-16)=2. Paralelno se računa intenzitet gradijenta (paljenjem prelaza t-6) prema formuli (5) i smer gradijenta (paljenjem prelaza t-7) prema formuli (6). Paralelnim paljenjem prelaza t-6 i t-7 menjaju se označavanja: μ(p-16)=0 i μ(p-17)= (p-18)=2. Mesta p-17 i p-18 modeluju izračunate intenzitete i smerove gradijenata, respektivno. Mesta p-19 i p-20 modeluju pragove *strong_th* i *weak_th* za određivanje jakih i slabih ivica. Sledećim paralelnim paljenjem skupa prelaza {t-8, t-9} paralelno se detektuju pikseli koji pripadaju jakim ivicama (paljenjem prelaza t-8) i prihvataju slabi pikseli koji su povezani sa jakim ivicama (paljenjem prelaza t-9). Paljenjem skupa prelaza {t-8, t-9} menjaju se označavanja: μ(p-17)=μ(p-18)=μ(p-19)=μ(p-20)=0 te μ(p-21)=μ(p-22)=1. Mesto p-21 modeluje piksele odabrane prema pripadnosti jakim ivicama, a mesto p-22 modeluje piksele koji pripadaju slabim ivicama koje su povezane sa jakim ivicama. Na kraju, paljenjem prelaza t-10 koje modeluje sinhronizaciju pri pravljenju unije rešenja navedenih piksela dosegnut je mrtav čvor gde je μ=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1). Mesto p-23 modeluje konačnu sliku koja predstavlja rezultat rada drugog nivoa predloženog algoritma.

U prethodno opisanom sledu izvršavanja datog UPN modela za dato početno označavanje pokazano je gde je moguće paralelno izvršavanje koda (skup prelaza koji se paralelno izvršavaju) te delovi UPN mreže koji obezbeđuju sinhronizaciju navedenog paralelizma.

Nakon modelovanja, simulacije i analize prikazani UPN model je transformisan u konkretan računarski program.

IV. Eksperimentalni rezultati

Korišćenjem računarskog programa koji je dobijen transformacijom datog UPN modela urađena je eksperimentalna primena predloženog algoritma nad skupom odabranih slika. Korišćene su dostupne slike sa različitih satelita. Sve slike korišćene u primerima su svedene na dimenzije 256×256.

Parametri transformacije na koje smo uticali smo predstavili kao dvanaestorku *pt*, pri čemu je sa *n*×*n* označena veličina kernela Gausovog zamućenja: *pt*=(k_r, k_g, k_b, min_i, max_i, min_o, max_o, *n*×*n*, *th_low, th_high, weak_th, strong_th).*

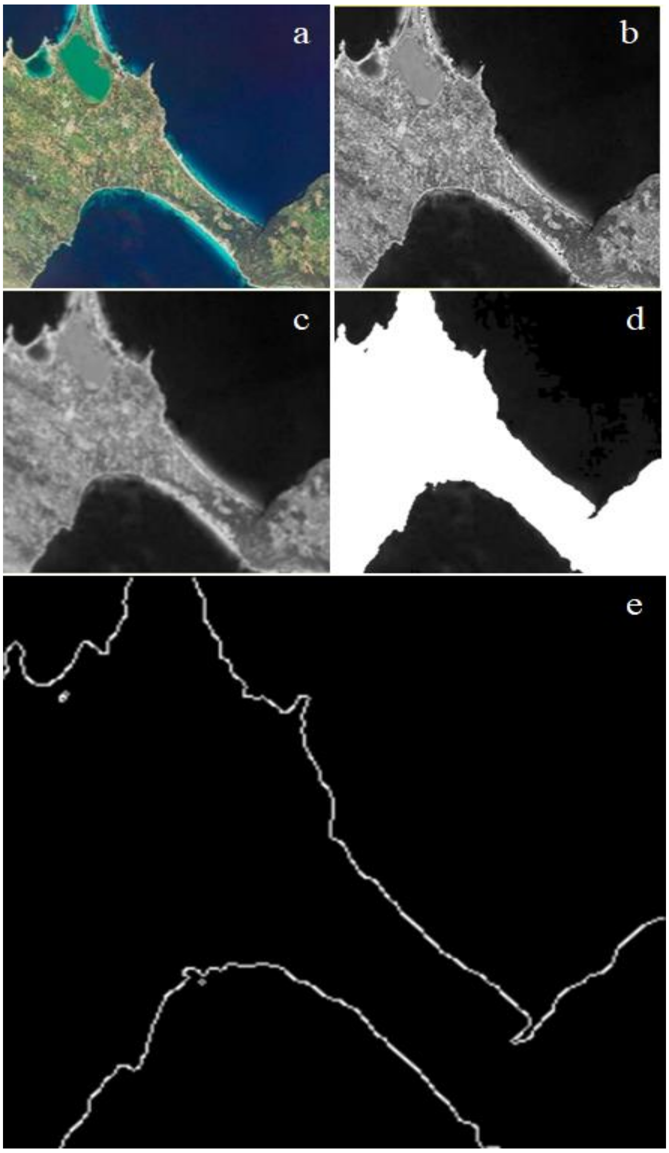
Svaka slika u primerima koji slede u naslovu sadrži vrednosti uređenih elemenata dvanaestorke *pt*.

U eksperimentima smo koristili vrednosti Gausovog zamućenja 3x3 ili 5x5, uz različite vrednosti dvostrukog praga odsecanja [8][9].

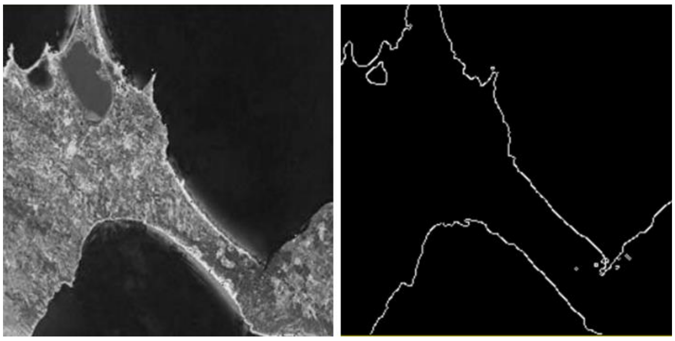
Na slici 3 dat je sled od 5 slika Balearskog ostrva Formentera kao što sledi: (a) originalna slika, (b) slika nakon konverzije u sivu sliku i razvlačenja kontrasta, (c) slika nakon primene Gausovog zamućenja, (d) slika nakon primene dvostrukog praga i (e) slika koja je rezultat drugog nivoa predloženog algoritma.

U ovom primeru, parametar k_g je podešen na maksimalnu vrednost 1.0. Parametri k_r i k_b imaju vrednost 0.0. Parametri za razvlačenje kontrasta imaju sledeće vrednosti: min_i=21, max_i=234, min_o=0, max_o=255. Kernel Gausovog zamućenja je veličine 3×3, a parametri dvostrukog praga imaju vrednost *th_low*=18 i *th_high*=67. Parametri drugog nivoa imaju vrednost *weak_th*=30 i *strong_th*=50.

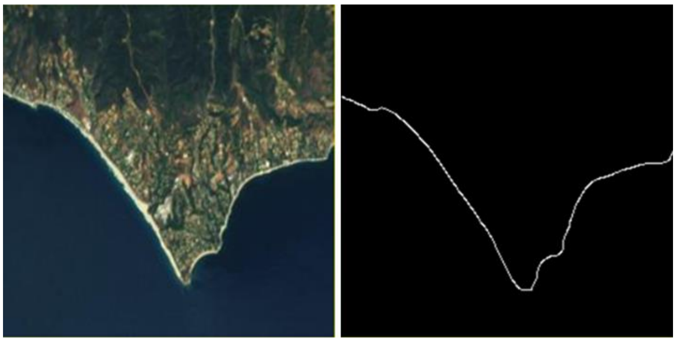
U primeru sa slike 4 algoritam je primenjen na istu sliku kao u primeru sa slike 3, ali sa drugačijim parametrima transformacije u sivu. Izabrane su ujednačene vrednosti parametara, k_r=0.333, k_g=0.334, k_b=0.333. Ostali parametri su isti kao u primeru sa slike 3. Uočljivo je da su rezultati lošiji nego u primeru sa slike 3. Lošiji rezultat se dobija i primenom podrazumevanih koeficijenata (k_r=0.299, k_g=0.587, k_b=0.114).



Sl. 3. Detekcija ivica na slici ostrva Formentera, Baleari, sat. Kanopus V1 *pt*=(0.0, 1.0, 0.0, 21, 234, 0, 255, 3×3, 18, 67, 30, 50)

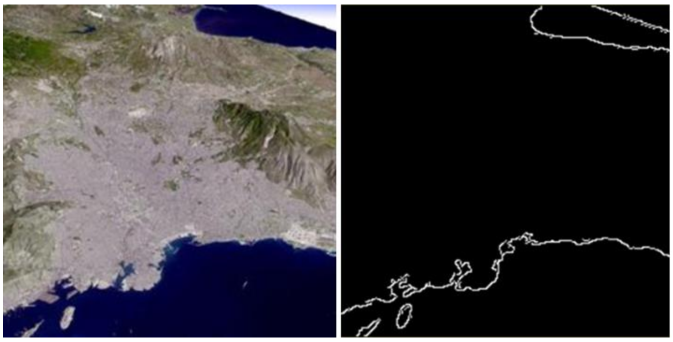


Sl. 4. Ostrvo Formentera, Baleari, satelit Kanopus V1, levo - transformacija u sivu i razvlačenje kontrasta, desno - detekcija ivica, *pt*=(0.333, 0.334,0.333, 21, 234, 0, 255, 3×3, 18, 67, 30, 50)



Sl. 5 Aerodrom Nica (satelit LANDSAT 7 ETM+), levo - originalna slika, desno - detektovana ivica, *pt*=(0.299, 0.587, 0.114, 0, 255, 0, 255, 5×5, 46, 238, 49, 81)

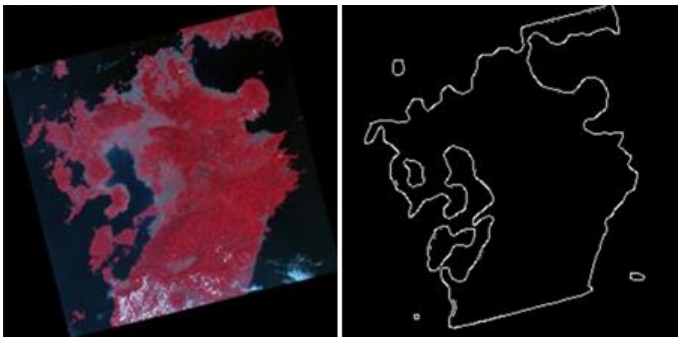
U primeru sa slike 5 primenjeni su podrazumevani parametri transformacije u sivu (k_r=0.299, k_g=0.587, k_b=0.114). Podešeni su parametri razvlačenja kontrasta min_i=0, max_i=255, min_o=0, max_o=255, a parametri dvostrukog praga *th_low*=46 i *th_high*=238, kao i parametri drugog nivoa *weak_th*=49 i *strong_th*=81. Najbolji rezultat je postignut sa kernelom Gausovog zamućenja veličine 5×5. Većim stepenom zamućenja ujednačeni su vrlo različiti tonovi zelene boje kopna, što je omogućilo detekciju kopna kao jedinstvene površine.



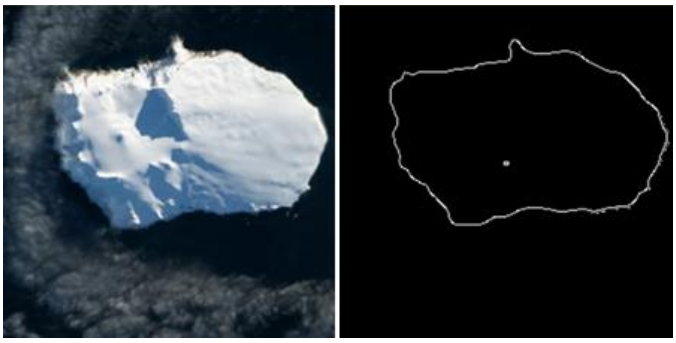
Sl.6 Atina (satelit Aster), levo - originalna slika, desno - detektovana ivica, *pt*=(0.420, 0.580, 0.0, 0, 255, 0, 255, 3×3, 41, 213, 35, 100)

Na slici 6 dat je primer gde je najbolji rezultat postignut sa parametrima transformacije u sivu k_r=0.420, k_g=0.580, k_b=0.0, parametrima razvlačenja kontrasta min_i=0, max_i=255, min_o=0, max_o=255, parametrima dvostrukog praga *th_low*=41 i *th_high*=213, kao i parametrima drugog nivoa *weak_th*=35 i *strong_th*=100, dok je kernel Gausovog zamućenja podešen na veličinu 3×3. Uočljivo je da je odgovarajućim izborom parametara transformacije u sivu, bez razvlačenja kontrasta, postignut najbolji rezultat.

U primeru sa slike 7 parametar k_r ima maksimalnu vrednost 1.0, a k_g i k_b vrednost 0.0. Podešeni su parametri razvlačenja kontrasta min_i=0, max_i=255, min_o=0, max_o=255, a parametri dvostrukog praga *th_low*=70 i *th_high*=224, kao i parametri drugog nivoa *weak_th*=31 i *strong_th*=50, dok je kernel Gausovog zamućenja podešen na veličinu 5×5. Plavo-crveni i belo-crveni delovi kopna su kernelom Gausovog zamućenja veličine 5×5 dodatno prigušeni, što ih je učinilo jedinstvenom površinom sa ostalim kopnom.

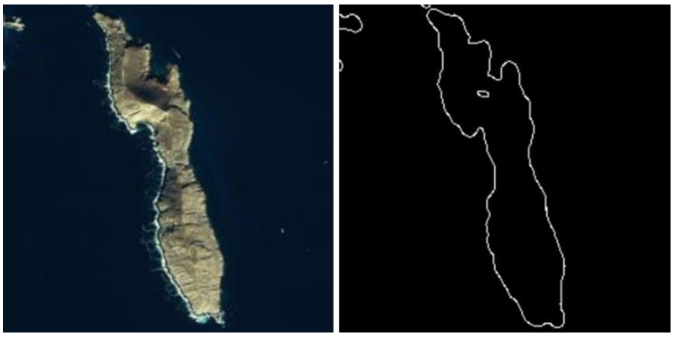


Sl. 7 Kjušu, Japan (sat. UK-DMC2), levo - originalna slika, desno - detektovana ivica, *pt*=(1.0, 0.0, 0.0, 0, 255, 0, 255, 5×5, 70, 224, 31, 50)



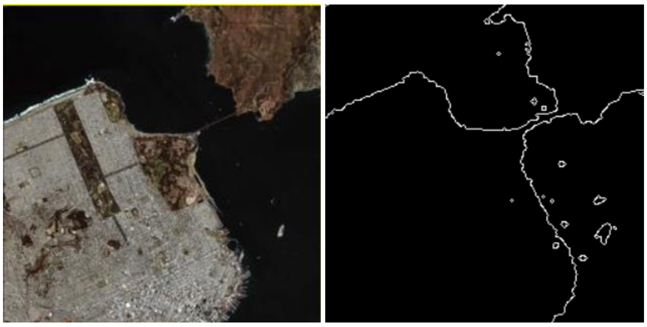
Sl. 8 Ostrvo Bouvet, Norveška (sat. Landsat 8), levo - originalna slika, desno - detektovana ivica, *pt*=(0.0, 0.0, 1.0, 5, 250, 0, 255, 5×5, 17, 126, 45, 50)

U primeru datom na slici 8 parametar k_b ima maksimalnu vrednost 1.0, a k_g i k_r vrednost 0.0. Podešeni su parametri razvlačenja kontrasta min_i=5, max_i=250, min_o=0, max_o=255, a parametri dvostrukog praga *th_low*=17 i *th_high*=126, kao i parametri drugog nivoa *weak_th*=45 i *strong_th*=50, dok je kernel Gausovog zamućenja podešen na veličinu 5×5. Sa datim parametrima razvlačenja kontrasta, dobijene su oštre ivice koje razdvajaju površine. Kernel Gausovog zamućenja veličine 5×5 je doprineo ujednačavanju kako zatamnjene površine senke sa ostatkom kopna, tako i ujednačavanju oblaka sa površinom mora, pa su spoljne ivice dobro detektovane.



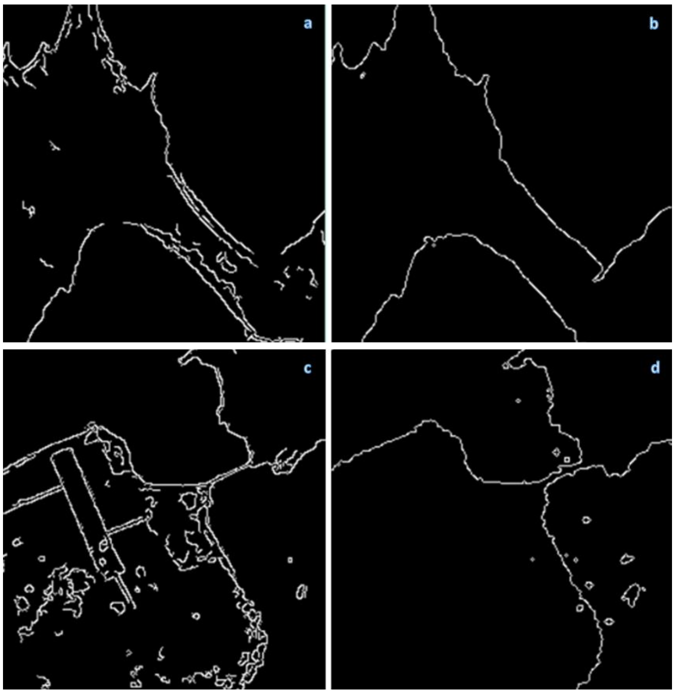
Sl. 9 Ostrvo Bajaca, Meksiko (sat. Quickbird), levo - originalna slika, desno - detektovana ivica, *pt*=(1.0, 0.0, 0.0, 1, 254, 0, 255, 5×5, 16, 27, 49, 49)

U primeru sa slike 9 jedan od boljih rezultata je postignut sa parametrima transformacije u sivu k_r=1.0, k_g=0.0, k_b=0.0, parametrima razvlačenja kontrasta min_i=1, max_i=254, min_o=0, max_o=255, parametrima dvostrukog praga *th_low*=16 i *th_high*=27, kao i parametrima drugog nivoa *weak_th*=49 i *strong_th*=49, dok je kernel Gausovog zamućenja podešen na veličinu 5×5.



Sl. 10 San Francisko (sat. Ikonos), levo - originalna slika, desno - detektovana ivica, *pt*=(1.0, 0.0, 0.0, 25, 207, 0, 255, 3×3, 211, 255, 30, 50)

U primeru sa slike 10 rezultat je postignut sa parametrima transformacije u sivu k_r=1.0, k_g=0.0, k_b=0.0, parametrima razvlačenja kontrasta min_i=25, max_i=207, min_o=0, max_o=255, sa kernelom Gausovog zamućenja veličine 3×3, parametrima dvostrukog praga *th_low*=16 i *th_high*=27, kao i parametrima drugog nivoa *weak_th*=49 i *strong_th*=49.



Sl. 11 Poređenje rezultata detekcije ivica za primere sa slika 3 i 10, Kanijevim algoritmom (slike a i c), te dvonivovskim algoritmom (slike b i d)

Na slici 11 prikazano je poređenje dobijenih rezultata detekcije ivica za primere date na slikama 3 i 10 upotrebom Kanijevog algoritma i dvonivovskog algoritma, respektivno.

V. Zaključak

Primenom dvonivovskog algoritma za detekciju ivica u eksperimentu na primeru fotografija dobijenih sa satelita pokazalo se da se predloženim postupkom mogu postići dobri rezultati u detektovanju ivica kopna i mora varijacijom različitih vrednosti parametara: za transformaciju slike u boji u pripadnu sivu sliku, za razvlačenje kontrasta sive slike, za Gausovo zamućenje, za dvostruki prag te za jake i slabe ivice kod odabiranja konačnih ivica. Tokom eksperimenta pojavljuju se problemi u detekciji, koji se odnose na razdvajanje realnih objekata na kopnu i moru (zgrade, plovni objekti, oblaci, senke visokih objekata i sl.) od ivica obale. Ovo implicira da u daljem razmatranju treba koristiti različite fotografije istih objekata, snimane iz različitih uglova, kako bi se efekti senki i oblaka minimizirali. Posebnim algoritmima treba prepoznati objekte i "utopiti" ih u strukturu kopna ili mora.

Literatura

[1] P. Upadhyay, S. Gupta, “Introduction to Satellite Imaging Technology and Creating Images using Raw Data Obtained from Landsat Satellite”, Interantional Journal of Global Technology Initiatives, vol. 1, issue 1, pp. 126-134, March 2012.

[2] P. Štrbac, G. Milovanović, “Upgraded Petri net model and analysis of adaptive and static arithmetic coding, Elsevier: Mathematical and Computer Modelling Vol. 58, pp. 1548–1562, 2013.

[3] J. Canny, “A Computational Approach to Edge Detection”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 679-698, November 1986.

[4] A. Güneş, H. Kalkan, E. Durmuş, “Optimizing the color-to-grayscale conversion for image classification”, Signal Image and Video Processing, vol. 10, pp. 863-870, October 2015.

[5] Zohair Al-Ameen, “Contrast Enhancement for Color Images Using an adjustable Contrast Stretching”, technique International Journal of Computing, vol. 17, no. 2, pp. 74-80, 2018.

[6] R. Liu, J. Mao, “Research on Improved Canny Edge Detection Algorithm”, MATEC Web of conferences no. 323, pp. 1-4, 2018.

[7] S. Ilkin, F. S. Hangisi, M. Tafrali, S. Sahin, “The Enhancement of Canny Edge Detection Algorithm Using Prewitt, Robert and Sobel Kernels, International Conference on Engineering Technologies (ICENTE'17), Konya, December 2017.

[8] Getreuer, Pascal, "A Survey of Gaussian Convolution Algorithms", IPOL (Image Processing On Line), December 2013.

[9] Y.K. Lai, P.L. Rosin, "Efficient Circular Thresholding", IEEE Trans. on Image Processing", vol 23, issue 3, pp. 992–1001, 2014

Application of two-level algorithm for edge detection modeled in Upgraded Petri-nets

Abstract — This work presents a two-level algorithm for edge detection. The first level includes a color image transformation into a gray image, stretching of contrast, application of Gaussian blur, and application of a double threshold. The second level refers to parts of the Canny algorithm: gradient calculation, non-maximum suppression, retention of strong-edge pixels, and weak-edge pixels associated with strong edges. The model of the proposed algorithm was modeled in Upgraded Petri nets (UPN) and then transformed into a specific computer program. The experimental results refer to the application of the proposed algorithm over the selected set of images with comparisons with the Canny algorithm are presented.

Keywords — edge detection; contrast stretching, Canny, Upgraded Petri-nets.

Perica Štrbac, Vukman Korać, Miloš Pejanović

Jedno rješenje proširenja PVR modula podrškom za više skladišnih uređaja

Radenko Banović, Ilija Bašičević

Apstrakt — **PVR (eng. Personal Video Recorder) je jedan od najvažnijih modula DTV (eng. Digital Television) prijemnika te je postao sastavni dio skoro svakog STB (eng. Set-top box) uređaja. U nekim državama poput Japana je zakonski zabranjeno vraćanje televizijskog sadržaja u nazad (eng. Timeshifting) pa je PVR iz tog razloga još značajniji. Na modernim STB uređajima je povećan broj USB otvora (eng. Ports), te svaki uređaj ima najmanje dva USB otvora čime se stvara mogućnost spajanja STB uređaja sa više skladišnih uređaja. Takođe, povećan je i lokalni skladišni prostor STB uređaja, te se pojavljuje mogućnost skladištenja DTV toka podataka i na lokalni skladišni prostor. U ovom radu je predstavljeno jedno rješenje proširenja PVR modula koji je sastavni dio DTV srednjeg sloja. Srednji sloj implementira standarde vezane za digitalnu televiziju i služi za obradu podataka na prijemnoj strani, kao i za apstrakciju kompleksnijih modula koji se nalaze ispod srednjeg sloja.**

Ključne reči — **Personal Video Recorder, Storage Devices, Digital Television, Middleware**

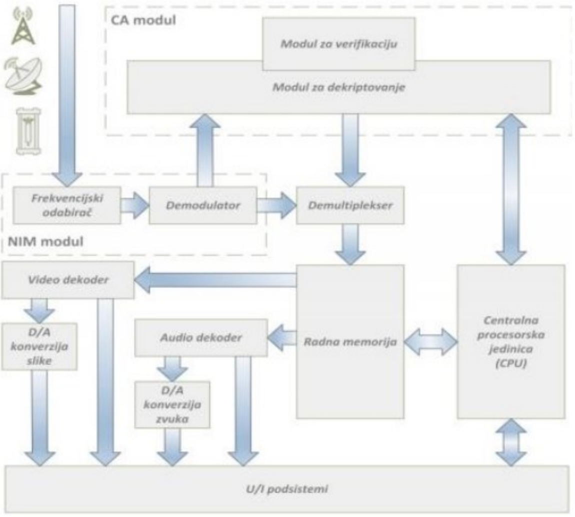
I. Uvod

Broj usluga koje televizija omogućuje je drastično promijenjen digitalizacijom televizijskog sadržaja. Televizija je služila za gledanje isključivo živog sadržaja bez mogućnosti manipulacije tokovima podataka ili gledanja propuštenog sadržaja. Digitalizacija je omogućila razvijanje novih funkcionalnosti kao što su DLNA (eng. Digital Living Network Alliance), PVR, EPG (eng. Electronic Program Guide), vraćanje televizijskog sadržaja unazad biranjem događaja iz elektronskog programskog vodiča, i mnogih drugih. Jedna od osnovnih funkcionalnosti modernih STB uređaja je PVR koji omogućuje snimanje televizijskog sadržaja na skladišni uređaj, gledanje snimljenog televizijskog sadržaja, pauziranje reprodukovanja, vraćanje u nazad snimljenog sadržaja, premotavanje itd.

Postoji nekoliko primjera implementacije PVR funkcionalnosti u okviru STB uređaja [1]-[5], ali su svi implementirani sa podrškom za jedan skladišni uređaj. Razlozi za to leže u činjenici da su lokalni skladišni prostori STB uređaja bili mali i STB uređaji su sadržali jedan USB otvor, te se na njega mogao povezati samo jedan spoljašni skladišni prostor. Set-top box uređaji danas imaju dovoljno velik lokalni skladišni prostor da se na njega mogu skladištiti televizijski tokovi podataka, te je broj USB otvora minimalno 2 što omogućuje povezivanje nekoliko spoljnih skladišnih uređaja. Time su stvoreni hardverski preduslovi za softversko proširenje PVR modula podrškom za više skladišnih uređaja.

II. Arhitektura PVR modula

PVR modul je sastavni dio DTV srednjeg sloja koji se izvršava na STB fizičkoj arhitekturi. Digitalna televizija je oblast koja je u velikom mjeri zavisna od fizičke arhitekture prijemnika i razvoj novih usluga zavisi od iskoraka u razvoju fizičke arhitekture. Primjer fizičke arhitekture STB uređaja je prikazan u Sl. 1.



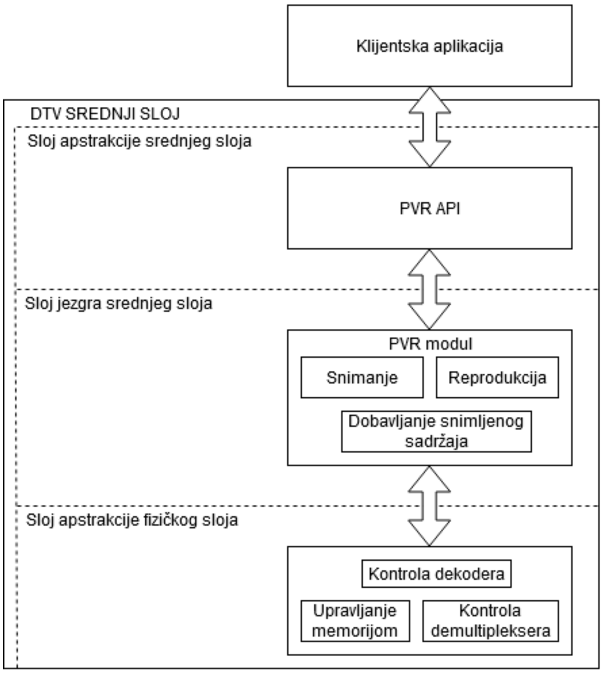
Sl. 1. Primjer fizičke arhitekture STB prijemnika

Srednji sloj (eng. Middleware) apstrahuje funkcionalnosti STB uređaja i obezbjeđuje potrebne programske sprege ka višim softverskim nivoima kao što je UI (eng. User Interface) korisnička aplikacija. Zahvaljujući ovom sloju viši softverski slojevi ne moraju poznavati specifičnosti vezane za digitalnu televiziju, već proslijeđivanjem osnovih podataka upravljaju nižim softverskim i hardverskim komponentama.

Softverska arhitektura srednjeg sloja sa prikazanim PVR blokovima je prikazana u Sl. 2. UI korisnička aplikacija koja se nalazi iznad PVR API (eng. Application Programming Interface) sloja šalje zahtjeve kao što su: zahtjev za započinjanje snimanja, zahtjev za dostavljanje informacija o dostupnom snimljenom sadržaju kao i zahtjev za reprodukovanjem snimljenog sadržaja. Nakon obrade zahtjeva, UI aplikacija od srednjeg sloja dobija podatke koje prikazuje korisniku.

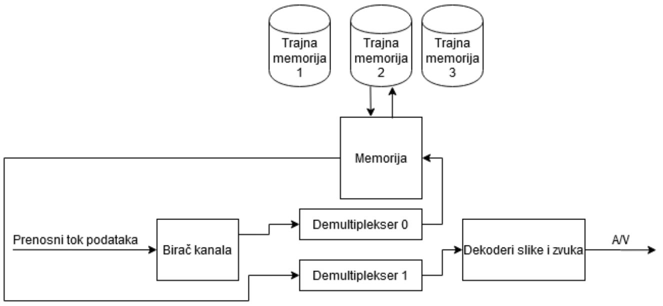
Radenko Banović – Fakultet Tehničkih Nauka, Trg Dositeja Obradovića 6,21000 Novi Sad, Srbija (e-mail:Radenko.Banovic@rt-rk.com).

Ilija Bašičević – Istrazivacko-razvojni Institut RT-RK, Novi Sad, Srbija (e-mail:Ilija.Basicevic@rt-rk.com).



Sl. 2. Softverska arhitektura srednjeg sloja sa akcentom na PVR blokove

PVR modul treba da obezbijedi različito usmjeravanje prenosnih tokova podataka između logičkih blokova srednjeg sloja u odnosu na prikazivanje podataka koji su tek prispjeli na birač kanala. Uprošteni dijagram usmjeravanja toka podataka između logičkih blokova srednjeg sloja u slučaju PVR snimanja i reprodukcije snimljenog sadržaja sa podrškom za više skladišnih uređaja je prikazan na Sl. 3. Prilikom PVR snimanja sadržaja sadržaj se smiješta u trajnu memoriju, dok se prilikom PVR reprodukcije sadržaj čita iz trajne memorije, usmjerava se na demultiplekser pa na dekodere slike i zvuka nakon čega se slika i zvuk prikazuju korisniku.



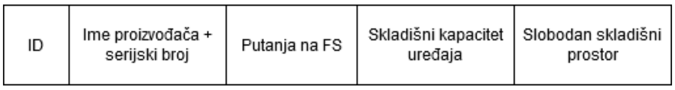
Sl. 3. Blok dijagram logičkih blokova sa podrškom za PVR

III. Opis realizacije

PVR modul je proširen tako da se korisničkoj aplikaciji u svakom trenutku mogu dostaviti skladišni uređaji dostupni za skladištenje kao i sav snimljen sadržaj na svim skladišnim uređajima priključenim na STB. Da bi tako nešto bilo moguće kreirane su dvije liste, prva sadrži dostupne uređaje dok druga sadrži dostupan snimljen sadržaj.

A. Lista dostupnih uređaja za skladištenje podataka

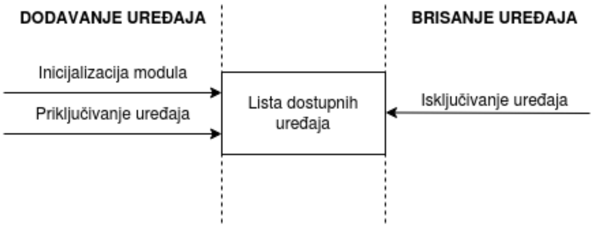
Svaki skladišni uređaj je opisan strukturom podataka koju čine jedinstveni identifikacioni broj svakog ugređaja, zatim ime proizvođača uređaja sa serijskim brojem, putanja na FS (eng. File System) na kojoj se nalazi uređaj, puni skladišni kapacitet uređaja, i na kraju slobodan skladišni prostor uređaja. Opis uređaja je prikazan u Sl. 4.



ID	Ime proizvođača + serijski broj	Putanja na FS	Skladišni kapacitet uređaja	Slobodan skladišni prostor

Sl. 4. Opis jednog skladišnog uređaja

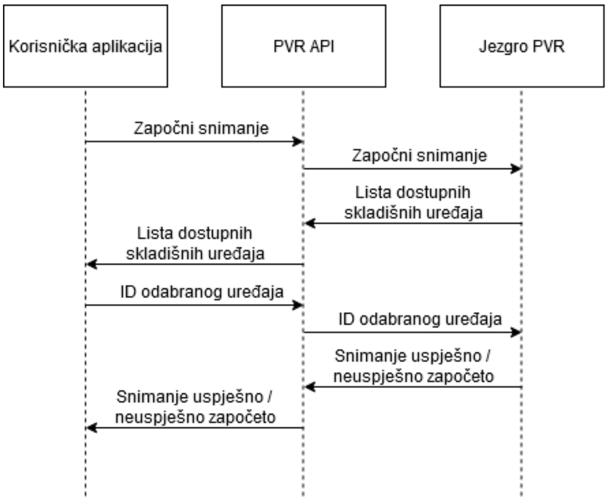
Na inicijalizaciji PVR modula se lokalni skladišni prostor STB uređaja dodaje u listu dostupnih uređaja, dok se spoljašni skladišni prostori u listi dodaju prilikom priključivanja skladišnih prostora na STB. Brisanje uređaja iz liste dostupnih uređaja se vrši prilikom isključivanja spoljašnjeg uređaja iz USB otvora. Dijagram koji opisuje dodavanje i brisanje uređaja iz liste je prikazan na Sl. 5.



Sl. 5. Dijagram dodavanja i brisanja uređaja iz liste

B. Započinjanje PVR snimanja

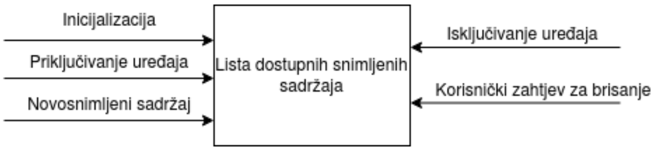
U proširenom PVR modulu koji podržava više skladišnih uređaja za započinjanje PVR snimanja neophodno je da korisnik izabere uređaj na koji želi da skladišti podatke. Stoga, prvi korak jeste dobavljanje svih dostupnih uređaja do aplikacije nakon čega korisnik bira jedan od ponuđenih uređaja, te se ID uređaja prosljeđuje zajedno sa ostalim podacima neophodnim za započinjanje PVR snimanja. Dijagram započinjanja PVR snimanja je prikazan na Sl. 6.


Korisnička aplikacija
PVR API
Jezgro PVR
Započni snimanje
Započni snimanje
Lista dostupnih skladišnih uređaja
Lista dostupnih skladišnih uređaja
ID odabranog uređaja
ID odabranog uređaja
Snimanje uspješno / neuspješno započeto
Snimanje uspješno / neuspješno započeto

Sl. 6. Dijagram započinjanja PVR snimanja

C. Lista dostupnih snimljenih sadržaja

Lista dostupnih snimljenih sadržaja sastoji se od snimljenih sadržaja sa svih skladišnih uređaja. Ona se popunjava prilikom inicijalizacije za sadržaj skladišten na STB uređaj, dok se prilikom priključivanja spoljašnjih uređaja dodaje sadržaj sa tih uređaja (prilikom inicijalizacije se priključeni spošaljni uređaji posmatraju kao ponovno priključeni). Takođe, u listu se dodaje novosnimljeni sadržaj, dok se iz liste izbacuje sadržaj snimljen na spoljašnji uređaj koji je isključen iz STB uređaja. Dijagram je prikazan na Sl 7.



Sl. 7. Dijagram dodavanja i brisanja sadržaja u i iz liste

IV. TESTIRANJE

Predloženo proširenje je testirano na nekoliko različitih platformi pri čemu je korišten različit broj skladišnih uređaja u zavisnosti od platforme. Platforme na kojima je rješenje testirano su Broadcom BCM 72180, HiSilicon Hi3796MV200 i AMLogic AML905. Provjera ispravnosti rada je vršena korištenjem komercijalnog alata BBT (eng. Black Box Testing) koji je namjenjen za tu svrhu. Eksterni skladišni uređaji koji su korišteni su Toshiba kapaciteta 1TB, Kingston kapaciteta 8 i 16 GB.

Automatskim testiranjem je moguće pokrenuti veliki broj testova čime se sa velikom vjerovatnoćom može verifikovati ponuđeno rješenje. Alat koji služi za automatsko testiranje se nalazi iznad PVR API sloja te se simulira zahtjev korisnika za snimanjem, reprodukcijom ili prikazom svih do sada snimljenih dostupnih sadržaja, te se povratne vrijednosti upoređuju sa željenim povratnim vrijednostima čime se potvrđuje valjanost. Takođe, moguće je simulirati priključenje ili isključenje spoljašnih skladišnih uređaja, te se velikim brojem automatskih poziva može sa velikom sigurnošću verifikovati rad modula.

Nakon nekoliko testnih kampanja u kojima je testiran PVR modul u cjelini, te analize dobijenih rezultata, pokazano je da su svi testovi bili uspješni, čime je proširenje i verifikovano.

V. ZAKLJUČAK

Postojeće PVR modul je proširen podrškom za više skladišnih uređaja čime je korisniku omogućeno kombinovanje više skladišnih uređaja manjeg kapaciteta čime je povećan kvalitet usluge vezane za PVR. Rješenje je testirano na nekoliko ciljnih platformi i utvrđena je funkcionalnost rješenja.

U budućnosti ovo rješenje može biti unaprijeđeno podrškom za skladišni uređaj u oblaku (eng. Cloud) čime bi se značajno povećao kvalitet usluge. Takođe, podrškom za skladištenje PVR sadržaja u oblaku otvara se mogućnost za daljim unaprjeđenjem usluge tako što bi korisnicima bilo omogućeno dijeljenje sadržaja na koji su svi pretplaćeni bez potrebe da svaki korisnik za sebe snima isti sadržaj.

ZAHVALNICA

Ovaj rad je delimično finansiran od strane Ministarstva za prosvetu, nauku i tehnološki razvoj Republike Srbije.

LITERATURA

[1] D. Sijakov, D. Sabo, M. Mirkovic, "One proposal of solution for Personal Video Recorder," Telecommunications Forum (TELFOR), Belgrade, Serbia, pp. 1003 - 1006, 26-28 Nov. 2013.

[2] M. Kovačević, G. Miljković, B. Korać, "The realization of a digital recorder function for STB based on Android," Telecommunications Forum (TELFOR), Belgrade, Serbia, pp. 1349 - 1352, 20-22 Nov. 2012.

[3] Personal Video Recorder, by M. Narasihman, R. O. Eifrig. (2017, Jun 20.). US20150319441A1. Accessed on: July. 13, 2020. [Online]. Available: https://patents.google.com/patent/US20150319441A1/en

[4] N. Fimic, S. Tanackovic, I. Basicevic, N. Teslic, "PVR trick-mode emulation through consecutive jumps," Telecommunications Forum (TELFOR), Belgrade, Serbia, 24-26 Nov. 2015.

[5] J. Bae, D. Kim, H. I. Kang, "An Efficient Personal Video Recorder System," 2010 International Conference on Intelligent Computation Technology and Automation

ABSTRACT

PVR (Personal Video Recorder) is one of the most important modules of DTV (Digital Television) receivers and has become an integral part of almost every STB (Set-top box) device. In some countries, such as Japan, timeshifting is legally forbidden, so PVR is even more important for that reason. On modern STB devices, the number of USB ports has increased, and each device has at least two USB ports, which creates the possibility of connecting STB devices with multiple storage devices. Also, the local storage space of the STB device has been increased, so the possibility of storing the DTV data stream to the local storage space also appears. This paper presents a solution for expanding a PVR module that is an integral part of the middle layer DTV. The middle layer implements the standards related to digital television and serves for data processing on the receiving side, as well as for the abstraction of more complex modules located below the middle layer.

One Solution of PVR Module Extension with Support for Multiple Storage Devices

Radenko Banović, Ilija Bašičević

Jedno rješenje poslužioca televizijskog sadržaja sa podrškom za IPv4 i IPv6 protokole

Radenko Banović, Ilija Bašičević

***Apstrakt* — U ovom radu je prikazano jedno rješenje poslužioca televizijskog sadržaja sa podrškom za IPv6 (Internet Protokol verzije 6) kao i IPv4 (Internet Protokol verzije 4) prenosom podataka višeznačnim i jednoznačnim upućivanjem i korištenjem RTP (eng. Real-Time Transport Protocol) / RTSP (eng. Real-Time Streaming Protocol) protokola. Zbog masovnog prelaska na IPv6 protokol pojavili su se prijemnici multimedijalnog toka podataka u kojima je implementiran isključivo IPv6 protokol iako je IPv4 (internet protokol verzije 4) i dalje najdominantniji internet protokol na svijetu. Krajnji cilj je da IPv6 protokol (čije se učešće u svjetskom internetu svakodnevno povećava) u konačnici u potpunosti zamijeni IPv4 protokol. Stoga, pojavila se potreba za poslužiocem koji podržava oba navedena protokola. Rješenje je testirano korištenjem nekoliko klijenata.**

***Ključne reči* — IPv4, IPv6, RTP, RTSP, IP**

I. Uvod

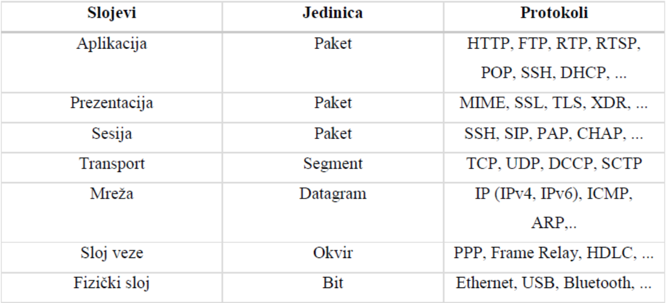
Dostavljanje televizijskog sadržaja putem interneta poslednjih godina doživljava veliki procvat zbog sve bolje infrastrukture računarskih mreža koja omogućuje brži protok podataka, što je prikazano u radovima [1] i [2]. Njegova prednost je u tome što omogućuje dvosmijernu klijent – server komunikaciju, te može da se koristi već postojeća mrežna infrastruktura, bez potrebe za pravljenjem nove fizičke infrastrukture.

Korištenje mrežne infrastrukture podrazumijeva implementaciju protokola koji služe za komunikaciju u računarskim mrežama. Da bi se TV (televizijski) sadržaj efikasno dostavljao putem interneta koristi se protokol za prenos u realnom vremenu RTP koji je enkapsuliran u UDP (eng. User Datagram protocol) i protokol za prenos toka u realnom vremenu RTSP. Sve dok traje tranzicijski period sa IPv4 na IPv6 protokol potrebno je imati rješenja koja podržavaju rad sa oba navedena protokola čime se omogućuje većem broju klijenata pristup željenom sadržaju.

Ovaj rad je sačinjen od 5 poglavlja. U prvom poglavlju su dat je kratak opis rada. U drugom dijelu su opisani protokoli korišteni u implementaciji. Opis rješenja je prikazan u trećem poglavlju u kom su opisane i ključne komponente sistema. U četvrtom poglavlju je opisan način funkcionalne provjere i opis testnih slučajeva. Peto poglavlje sadrži kratak pregled rada i potencijalna unaprjeđenja realizovanog rješenja.

II. Protokoli korišteni u implementaciji

Protokoli prikazani u ovom radu se najlakše vizuelno predstavljaju OSI referentnim modelom koji dijeli arhitekturu mreže u sedam logičkih nivoa, daje spisak servisa i protokola koji funkcionišu na svakom od nivoa (Sl. 1.).



Slojevi	Jedinica	Protokoli
Aplikacija	Paket	HTTP, FTP, RTP, RTSP, POP, SSH, DHCP, ...
Prezentacija	Paket	MIME, SSL, TLS, XDR, ...
Sesija	Paket	SSH, SIP, PAP, CHAP, ...
Transport	Segment	TCP, UDP, DCCP, SCTP
Mreža	Datagram	IP (IPv4, IPv6), ICMP, ARP,..
Sloj veze	Okvir	PPP, Frame Relay, HDLC, ...
Fizički sloj	Bit	Ethernet, USB, Bluetooth, ...

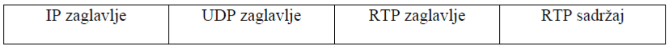
Sl. 1. Slojevi OSI modela

Tri gornja sloja (aplikacija, prezentacija i sesija) imaju ulogu da opišu proces komunikacije aplikacija među sobom kao krajnjim tačkama, dok donja četiri sloja (transport, mreža, sloj veze i fizički sloj) imaju ulogu definisanja prenosa informacije sa jednog na drugi kraj mreže.

A. RTP protokol

Protokol za prenos u realnom vremenu – RTP je mrežni protokol za dostavljanje audio i video sadržaja preko IP mreže. Najčešće se izvršava korišćenjem UDP protokola (Sl. 2.), kome je brzina prenosa podataka bitnija od kvaliteta primljenog signala.

Zbog grešaka u prenosu paketi mogu da se izgube, zakasne ili prenesu bez poštovanja redoslijeda. Protokol za prenos u realnom vremenu ne obezbjeđuje isporuku podataka na vrijeme i ne uključuje vremenska odstupanja. Međutim on u svom zaglavlju nosi podatke koji omogućavaju da se utvrdi koje su poruke izgubljene u prenosu i na osnovu toga ispravlja redoslijed paketa na prijemu. Sam RTP protokol ne može da rezerviše resurse na mreži, ili da obezbijedi kvalitet prenosa. RTP protokol je specificiran u dokumentu [3].



IP zaglavlje	UDP zaglavlje	RTP zaglavlje	RTP sadržaj

Sl. 2. Struktura IP paketa

Digitalizovan audio i video signal (u bitima) se smješta u RTP sadržaj ispred koga se dodaje RTP zaglavlje (Sl. 3.) koje nosi informacije o korisničkom sadržaju, načinu kodovanja, itd.

Radenko Banović – Fakultet Tehničkih Nauka, Trg Dositeja Obradovića 6, 21000 Novi Sad, Srbija (e-mail: Radenko.Banovic@rt-rk.com).

Ilija Bašičević – Istrazivačko-razvojni Institut RT-RK, Novi Sad, Srbija (e-mail: Ilija.Basicevic@rt-rk.com).

Redni broj bita	0-1	2	3	4-7	8	9-15	16-31
0	Verzija	P	X	CC	M	PT	Redni broj poruke
32	Vrijeme nastanka						
64	Identifikator izvora						
96	Lista učesnika						
96+32+CC	Definicija učesnika						Dužina zaglavlja proširenja
128+32+CC	Zaglavlje proširenja						

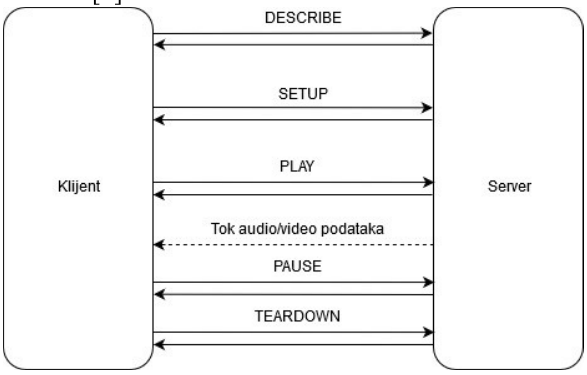
Sl. 3. Zaglavlje RTP paketa

B. RTSP protokol

Protokol za kontrolu prenosa toka u realnom vremenu (eng. Real-Time Streaming Protocol) je mrežni protokol dizajniran za kontrolu nad dostavljanjem audio/video sadržaja u realnom vremenu, na aplikativnom nivou. Ovaj protokol omogućava odabir kanala isporuke (kao i protokol dostavljanja sadržaja), dakle protokol za dostavljanje audio/video sadržaja ne mora biti RTP, već bilo koji drugi protokol sa istom namjenom (kao što je HLS (eng. Http Live Streaming).

Protokol se koristi za uspostavljanje i kontrolu veze između klijenta i servera, kao i slanje klijentskih zahtjeva kao što su puštaj (eng. Play), snimaj (eng. Record) i pauziraj (eng. Pause), što je prikazano u Sl. 4.

RTSP je baziran na RTSP zahtjevima koji se šalju od klijenta ka serveru, na koje server treba da odgovori i izvrši akciju koja se od njega očekuje. RTSP protokol je specificiran u dokumentu [4].



Sl. 4. Dijagram RTSP komunikacije

Svaki od RTSP zahtjeva i odgovora ima svoj format, primjer RTSP options zahtjeva je prikazan u Sl 5.

Zahtjev od strane klijenta	`OPTIONS rtsp://example.com/media.mp4RTSP/1.0`
	`CSeq: 1`
	`Require: implicit-play`
	`Proxy-Require: gzipped-messages`

Odgovor od strane poslužioca	`RTSP/1.0 200 OK`
	`CSeq: 1`
	`Public: DESCRIBE, SETUP, TEARDOWN, PLAY, PAUSE`

Sl. 5. RTSP options zahtjev

C. IP protokol

Internet protokol je protokol trećeg sloja OSI referentnog modela – sloja mreže. Sadrži informacije o adresiranju, čime se postiže da svaki mrežni uređaj (računar, server, radna stanica, ...) koji je povezan na internet ima jedinstvenu adresu i može se lako identifikovati u cijeloj internet mreži.

IP ima tri osnove funkcije:

- Adresiranje – definiše način dodjele internet adresa
- Rutiranje – određivanje putanje za prenos podataka sa jednog računara na drugi bez prethodne uspostave veze
- Fragmentaciju – i ponovno sastavljanje paketa kada je to potrebno

D. IPv4

Internet protokol verzija 4 je najrašireniji internet protokol na internetu, pojavljuje se 1983. godine, a definisan je u dokumentu RFC 791 septembra 1981. godine. IPv4 koristi 32-bitni adresni prostor što ga limitira na 2^{32} jedinstvenih adresa, ali veliki blokovi su rezervisani za specijalne mrežne metode (privatne mreže i adrese za višeznačno upućivanje).

IPv4 adresa može biti predstavljena bilo kojom notacijom koja predstavlja 32-bitnu cjelobrojnu vrijednost, ali je najčešće predstavljena sa po četiri 8-bitna bloka predstavljena decimalno, i odvojena tačkama [5].

E. IPv6

Pošto su zalihe IPv4 adresa pri samom kraju, pojavila se potreba za prelaskom na adrese sa većom adresnom širinom. IPv6 je novi, ali još ne široko korišćen, standardni internet protokol, čije su adrese 128 bita široke, što znači da postoji tačno 2^{128} unikatnih adresa, a to bi trebalo da zadovolji blisku budućnost. IPv6 adresa je predstavljena sa osam četvorocifrenih heksadecimalnih brojeva (8 puta po 16 bita) odvojenih dvotačkama. IPv6 ne uključuje mehanizam kontrolne sume [6]. Zaglavlje IPv6 paketa je prikazano u Sl. 6.

Verzija (4 bita)	Klasa saobraćaja (8 bita)	Oznaka toka (20 bita)	
Dužina podataka (16 bita)		Sledeće zaglavlje (8 bita)	Broj skokova (8 bita)
Adresa izvora (128 bita)			
Adresa odredišta (128 bita)			
Podaci dodatnog zaglavlja (promjenljiva dužina)			

Sl. 6. Zaglavlje IPv6 paketa

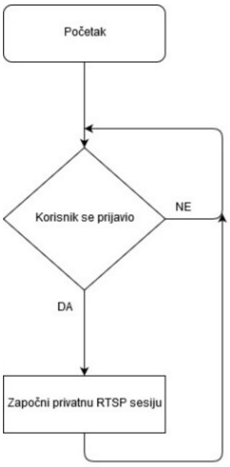
III. Opis rješenja

Ovaj poslužilac je realizovan tako da omogući i video na zahtjev posluženje podataka, kao i prenos podataka višeznačnim upućivanjem za živi prenos multimedijalnog sadržaja. Podešavanjem utičnica pruža se podrška za IPv4 I IPv6 protokole.

Izvršava se u komandnoj liniji (bez grafičkog interfejsa), osnovne informacije (port, putanja do fajla za prenos višeznačnim upućivanjem, itd.) se podešavaju proslijeđivanjem parametara prilikom pokretanja softvera, dok klijent može uticati na vrstu sadržaja koja mu se dostavlja korišćenjem RTSP protokola.

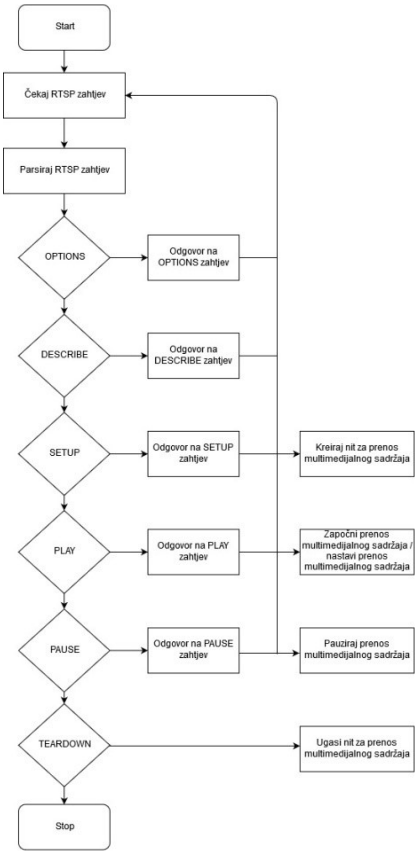
A. RTP/RTSP poslužilac jednoznačnim upućivanjem

Mora da implementira protokol koji se koristi za uspostavljanje u kontrolu veze između klijenta i poslužioca, kao i slanje klijentskih zahtjeva. Pošto je protokol za kontrolu prenosa toka u realnom – RTSP protokol, najkorišćeniji protokol za takvu namjenu, on će biti korišćen za kontrolu veze i slanje klijentskih zahtjeva između klijenta i poslužioca. Naredna slika prikazuje arhitekturu osnovne niti poslužioca.



Sl. 7. Osnovna arhitektura poslužioca jednoznačnim upućivanjem

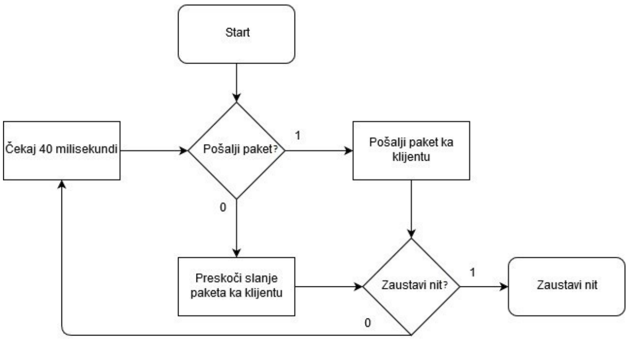
Osnovna nit čeka prijavljivanje korisnika i za svakog od prijavljenih korisnika pokreće posebnu nit koja vrši RTSP komunikaciju sa klijentom individualno koja je prikazana na na Sl. 8, bez ometanja drugih klijenata koji su istovremeno opsluživani od strane istog poslužioca.


Start
Čekaj RTSP zahtjev
Parsiraj RTSP zahtjev
OPTIONS
Odgovor na OPTIONS zahtjev
DESCRIBE
Odgovor na DESCRIBE zahtjev
SETUP
Odgovor na SETUP zahtjev
Kreiraj nit za prenos multimedijalnog sadržaja
PLAY
Odgovor na PLAY zahtjev
Započni prenos multimedijalnog sadržaja / nastavi prenos multimedijalnog sadržaja
PAUSE
Odgovor na PAUSE zahtjev
Pauziraj prenos multimedijalnog sadržaja
TEARDOWN
Ugasi nit za prenos multimedijalnog sadržaja
Stop

Sl. 8. Klijentska nit za RTSP komunikaciju

Prilikom obrade klijentskog RTSP zahtjeva OPTIONS potrebno je kreirati novu nit koja čita bitske vrijednosti iz MPEG-TS fajla koji će se korišćenjem RTP protokola prenositi do klijenta. Prilikom kreiranja niti kao parametar se prenose klijentski RTP i RTSP port, kao i identifikacioni broj koji jednoznačno određuje svakog klijenta.

U toku obrade RTSP zahtjeva PLAY i PAUSE mijenja se zastavica koja ima vrijednost 1 kada treba da se prenosi multimedijalni tok ka klijentu, odnosno 0 kada prenos treba da bude pauziran. Prilikom obrade RTSP zahtjeva TEARDOWN zastavica koja označava gašenje niti u kojoj se prenosi multimedijalni tok ka klijentu se postavlja na 1, što označava gašenje niti. Dijagram niti za prenos podataka je prikazan u Sl. 9.

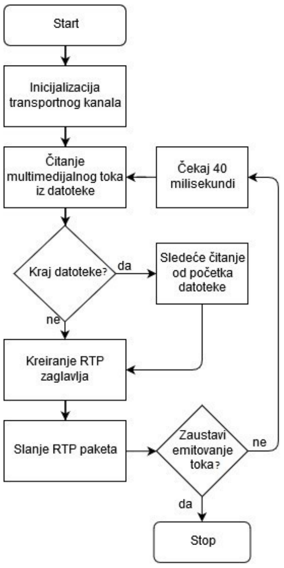


Sl. 9. Dijagram niti za prenos podataka

B. RTP poslužilac višeznačnim upućivanjem

Višeznačno upućivanje je napogodniji način prenosa multimedijalnih tokova koji se u jednom trenutku trebaju poslati ka nekoliko krajnjih korisnika, jer se u tom slučaju šalje samo jedan IP paket do grupe klijenata, čime se štedi propusni opseg a krajnjim korisnicima omogućava jednak kvalitet dostavljenog sadržaja.

Za ovakav vid prenosa podataka klijent treba da zna samo IP adresu poslužioca, port na koji će se emitovati multimedijalni tok podataka i protokol po kom se podaci šalju, te je njegova arhitektura dosta jednostavnija od arhitekture RTP/RTSP poslužioca jednoznačnim upućivanjem. Dijagram RTP poslužioca višeznačnim upućivanjem je prikazan u Sl. 10.

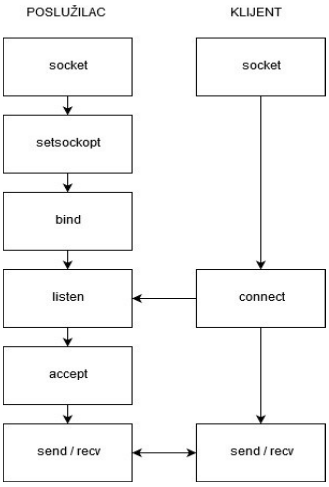

Start
Inicijalizacija transportnog kanala
Čitanje multimedijalnog toka iz datoteke
Čekaj 40 milisekundi
Kraj datoteke?
da
Sledeće čitanje od početka datoteke
ne
Kreiranje RTP zaglavlja
Slanje RTP paketa
Zaustavi emitovanje toka?
ne
da
Stop

Sl. 10. Dijagram RTP poslužioca višeznačnim upućivanjem

C. Podešavanje mrežnih utičnica

Mrežna utičnica (eng. Socket) je krajnja tačka dvosmijernog međuprocesnog komunikacionog toka preko računarske mreže bazirane na internet protokolu, kao što je internet. Adresa mrežne utičnice je spoj IP adrese i porta u jedinstven entitet. Mrežna utičnica je određena jedinstvenom kombinacijom protokol (TCP ili UDP), lokalne adrese mrežne utičnice (adresa i port) i udaljene adrese mrežne utičnice (samo za uspostavljene TCP mrežne utičnice).

Da bi mogli raditi sa mrežnim utičnicama u C++ programskom jeziku na Linux operativnom sistemu potrebno je uključiti datoteke <*sys/socket.h*>, <*netinet/in.h*> i <*arpa/inet.h*>. Dijagram komunikacije i kreiranja mrežnih utičnica je prikazan u Sl. 11.



Sl. 11. Dijagram komunikacije i kreiranja mrežnih utičnica

Prvi korak pri kreiranju mrežne utičnice je poziv funkcije *socket(int domain, int type, int protocol)* čija je povratna vrijednost file descriptor integer tipa koji jednoznačno određuje mrežnu utičnicu. Parametar domain određuje porodicu protokola koja će biti korištena za komunikaciju. Vrijednost koja se proslijeđuje za IPv6 protokol u parametru domain je AF_INET6, dok se za IPv4 protokol proslijeđuje vrijednost AF_INET. Parametar type određuje tip veze na transportnom nivou OSI model (TCP ili UDP). Prvi korak u ostvarivanju komunikacije sa klijentom je zasnivanje veze bazirane na RTSP protokolu, potrebno je otvoriti TCP vezu (koja je dvosmijerna i pouzdana), tako da se proslijeđuje vrijednost SOCK_STREAM. Poslednji parametar određuje koji protokol će biti korišten sa proslijeđenom porodicom protokola, tako da se najčešće proslijeđuje vrijednost 0.

Pozivom funkcije *setsockopt(int socket, int level, int option_name, const void *option_value, socklen_t option_len)* moguće je podesiti specifičnosti, ali nije obavezno. Kada je mrežna utičnica kreirana pozivom funkcije *socket()* ona još uvijek nema dodjeljenu adresu. Pozivom funkcije *bind(int sockfd, const struct sockaddr *addr, socklen_t addrlen)* dodjeljuje se adresa određena parametrom addr mrežnoj utičnici čiji je file descriptor proslijeđen kao parametar *sockfd*. Struktura koja se popunjava za IPv6 mrežu je sockaddr_in6, u polje *sin6_family* se postavlja vrijednost AF_INET6, u polje *sin6_port* se postavlja vrijednost koja je proslijeđena kao parametar komandne linije prilikom pokretanja aplikacije, dok se u polje *sin6_addr* postavlja vrijednost *in6addr_any*. Struktura koja se popunjaava za Ipv4 mrežu je *sockaddr_in* sa poljima *sin_family*, *sin_port*, *sin_addr*.

Pozivom funkcije *listen(int sockfd, int backlog)* označavamo mrežnu utičnicu određenu vrijednošću *sockfd* kao pasivnu mrežnu utičnicu, te će ona biti korištena za primanje zahtjeva za konekciju. Vrijednost parametra *backlog* određuje maksimalan broj neobrađenih zahtjeva za konekciju koji mogu biti prihvaćeni u jednom trenutku, svi koji pošalju zahtjev ako je broj neobrađenih zahtjeva popunjen dobiće poruku o grešci.

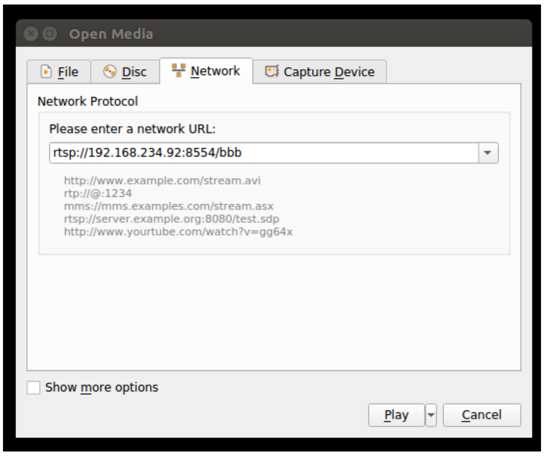
Funkcija *accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen)* je blokirajuća funkcija. Ona iz reda zahtjeva za konekciju izuzima prvi zahtjev u redu po FIFO (eng. First In First Out) redoslijedu, te kreira novu mrežnu utičnicu. Povratna vrijednost ove funkcije je file descriptor novokreirane mrežne utičnice koja predstavlja klijenta. Nakon ovoga poslužilac i klijent mogu da razmjenjuju poruke po IPv6 protokolu pozivanjem funkcija *send*() i *recv().*

IV. TESTIRANJE

Za testiranje je korišteno nekoliko klijenata: AVPlay, VLC, komercijalna aplikacija koja može da se izvršava na privatnom računaru i set-top box uređaju, kao i klijent napravljen u C programskom jeziku koji simulira rad klijenta i upisuje dobijene podatke u datoteku (Sl. 12 i Sl. 13).

A. Testiranje korišćenjem AVPlay i VLC klijenta

AVPlay i VLC klijenti su korišteni za testiranje RTP/RTSP poslužioca jednoznačnim upućivanjem.



Network Protocol

Please enter a network URL:

rtsp://192.168.234.92:8554/bbb

```
http://www.example.com/stream.avi
rtp://@:1234
mms://mms.examples.com/stream.asx
rtsp://server.example.org:8080/test.sdp
http://www.yourtube.com/watch?v=gg64x
```

Sl. 12. URL za uspostavljanje veze sa poslužiocem

Testiranja su vršena na način da se u klijentskoj aplikaciji unese URL koji je sadržan od protokola koji se koristi, IP adrese računara na kom se pokreće poslužilac (internet protokol verzije 4 ili verzije 6), porta, dok poslednji dio predstavlja sadržaj koji klijent želi da mu poslužilac dostavi.

B. Testiranje korišćenjem komercijalnog STB uređaja

Korišćenjem komercijalnog STB uređaja testirani su i poslužioc višeznačnim upućivanjem kao i poslužioc jednoznačnim upućivanjem.

Sl. 13. Prikazan sadržaj na komercijalnom STB uređaju

Klijent je prilagođen tako da mu je na jednom IPTV servisu promijenjena izvorna adresa poslužioca, dok je poslužilac slao podatke na adresu za višeznačno usmjeravanje koju klijent očekuje. Testiranje na klijentu je vršeno audio-vizuelno, i sve datoteke su se uspješno reprodukovale na klijentskoj strani.

C. Testiranje korišćenjem klijentske testne aplikacije

Razvijena je testna aplikacija koja u potpunosti simulira rad klijenta, tako da podržava i prenos podataka jednoznačnim i višeznačnim upućivanjem, RTP i RTSP protokol za IPv4 i IPv6 protokol, ali tako da umjesto prikaza audio i video sadržaja korisniku isti upisuje u izlaznu datoteku [7]. Prilikom testiranja poslužioca jednoznačnim upućivanjem zaustavljan je prenos podataka slanjem klijentske RTSP poruke Pause, i nastavljan je prenos podataka slanjem klijentske poruke Play. Izvorne datoteke sadrže multimedijalni sadržaj u TS (eng. Transport Stream) formatu. Izlazna datoteka bi po završetku prenosa podataka trebala da ima isti sadržaj kao izvorna datoteka da bi test bio uspješan. Poređenja su vršena korištenjem cmp komande Linux operativnog sistema koja poredi svaki bajt ulazne datoteke sa bajtom izlazne datoteke na istoj poziciji i prijavljuje prvu poziciju na kojoj se nalazi razlika u bajtima. Ukoliko su dve datoteke identične, komanda se završava bez ispisa o razlikama. Sva poređenja ulaznih i izlaznih datoteka su bila bez ispisa o greškama čime se potvrđuje da su testovi uspješni (Tabela I).

TABELA I
REZULTATI TESTOVA POSLUŽIOCA JEDNOZNAČNIM I VIŠEZNAČNIM UPUĆIVANJEM

Izvorna datoteka	Vrsta upućivanja/ Protokol	Različit ih bajta	Rezultat
izvorna 1.ts	Jednoznačno/ IPv4	0	Uspješan
izvorna 1.ts	Jednoznačno/ IPv6	0	Uspješan
izvorna 2.ts	Jednoznačno/ IPv4	0	Uspješan
izvorna 2.ts	Jednoznačno/ IPv6	0	Uspješan
izvorna.ts	Jednoznačno/ IPv4	0	Uspješan
izvorna.ts	Jednoznačno/ IPv6	0	Uspješan
izvorna 1.ts	Višeznačno/ IPv4	0	Uspješan
izvorna 1.ts	Višeznačno/ IPv6	0	Uspješan
izvorna 2.ts	Višeznačno/ IPv4	0	Uspješan
izvorna 2.ts	Višeznačno/ IPv6	0	Uspješan
izvorna.ts	Višeznačno/ IPv4	0	Uspješan

V. ZAKLJUČAK

U ovom radu je realizovan poslužilac multimedijalnog sadržaja sa podrškom za prenos podataka jednoznačnim i višeznačnim upućivanjem, i oba IP protokola. Opisana je arhitektura kompletne aplikacije, te je dat spisak funkcionalnosti. Do detalja je opisano kreiranje mrežnih utičnica tako da podržavaju rad sa oba IP protokola.

Rješenje je verifikovano na nekoliko klijentskih aplikacija, i krajnje rješenje je uspješno izvršilo sve testove. Dato rješenje je moguće nadograditi na nekoliko načina. Ova aplikacija podržava rad isključivo sa RTP protokolom, dok bi se mogla proširiti tako da podržava i neke druge protokole, poput HLS protokola. Time bi se proširio broj klijenata kojima bi multimedijalni sadržaj bilo moguće dostaviti.

Moguće je proširiti podršku i za ostale RTSP zahtjeve, kao što su GET_PARAMETER, SET_PARAMETER, REDIRECT, ANNOUNCE, itd.

ZAHVALNICA

Ovaj rad je delimično finansiran od strane Ministarstva za prosvetu, nauku i tehnološki razvoj Republike Srbije.

LITERATURA

[1] C.H. Lee, H.C. Chiu, “Telco IPTV growth strategy in Taiwan,” in *13th International Conference on Advanced Communication Tehchnology*, pp. 1469-1474 , Feb. 2011.

[2] E. H. Khabbiza, R. E. Alami, H. Qjidaa, “A new solution to optimize the time shift TV bandwidth,” in *International Conference on Intelligent Systems and Computer Vision (ISCV)*, Fez, Morocco, April, 2018.

[3] RTP: A Transport Protocol for Real-Time Applications, Jul 2003. [online]. Available: *https://tools.ietf.org/html/rfc3550*

[4] Real Time Streaming Protocol (RTSP), April 1998. [online]. Available: *https://tools.ietf.org/html/rfc2326*

[5] Internet Protocol, Septembar 1981. [online]. Available: *https://tools.ietf.org/html/rfc791*

[6] Internet Protocol Version 6 (IPv6), Decembar 1998. [online]. Available: *https://tools.ietf.org/html/rfc2460*

[7] Apparatus and method for dtv closed-captioning processing in broadcasting and communication system, by JeHo Nam. (2011, Jun 23). Accessed on Sept. 2. 2020. [Online]. Available: https://patents.google.com/patent/US20110149153A1/en

ABSTRACT

This paper presents a solution of television content streamer with support for IPv6 (Internet Protocol version 6) as well as IPv4 (Internet Protocol version 4). Unicast and multicast transmission is used and rtp/rtsp protocol. Due to the mass transition to IPv6 protocol, some media players have implemented only IPv6 protocol eventhough IPv4 is still the most dominant Internet protocol in the world. The ultimate goal is that the IPv6 protocol (whose participation in the global Internet is increasing daily) will eventually completely replace the IPv4 protocol. Therefore, there is a need for streamer that supports both of these protocols. The solution was tested using several clients.

One TV Content Server Solution with Support for IPv4 and IPv6 Protocols

Radenko Banović, Ilija Bašičević