Mobile robot decision-making system based on deep machine learning

Aleksandar Jokić, Milica Petrović and Zoran Miljković

***Abstract*—One of the major aspects of Industry 4.0 is enabling the manufacturing entities to operate in the dynamical systems autonomously. Therefore, to be autonomous, manufacturing entities need to have sensors to perceive their environment and utilize that information to make decisions regarding their actions. Having that in mind, in this paper, the authors propose a mobile robot decision-making system based on the integration of visual data and mobile robot pose. Mobile robot pose (current position and orientation) is integrated with two images gathered by two cameras and utilized to predict the possibility of gripping the part to be manufactured. A decision-making system is created by utilizing the deep learning model Resnet18 with an additional input for the mobile robot pose. The model is trained end-to-end and experimental evaluation is performed by using the mobile robot RACIO (Robot with Artificial Intelligence based COgnition).**

***Index Terms*—Decision-making system, mobile robots, deep learning.**

I. Introduction

Enabling mobile robots to operate in the manufacturing environment autonomously represents one of the fundamental requirements regarding Industry 4.0 concepts [1]. To fulfill this requirement, mobile robots need to localize themselves within the environment and use sensors to perceive the current state of the manufacturing system. In this paper, the authors propose to include both visual information and mobile robot pose in the make-decision process regarding future mobile robot actions. Mobile robot pose (Fig. 1), represented by position (x and y) and orientation (θ), is combined with image data to make a decision regarding the probability of successful gripping of the manufacturing part. The mobile robot's task is to move relatively close to the machine (marked with red color in Fig. 1) and decide if the current pose is adequate for a part (presented with a blue color in Fig. 1) picking process.

Aleksandar Jokić, teaching assistant, University of Belgrade - Faculty of Mechanical Engineering, Department of Production Engineering, Laboratory for industrial robotics and artificial intelligence (ROBOTICS&AI), Kraljice Marije 16, 11120 Belgrade 35, The Republic of Serbia (ajokic@mas.bg.ac.rs).

Dr. Milica Petrović, Associate Professor, University of Belgrade - Faculty of Mechanical Engineering, Department of Production Engineering, Laboratory for industrial robotics and artificial intelligence (ROBOTICS&AI), Kraljice Marije 16, 11120 Belgrade 35, The Republic of Serbia (mmpetrovic@mas.bg.ac.rs).

Dr. Zoran Miljković, Full Professor, University of Belgrade - Faculty of Mechanical Engineering, Department of Production Engineering, Laboratory for industrial robotics and artificial intelligence (ROBOTICS&AI), Kraljice Marije 16, 11120 Belgrade 35, The Republic of Serbia (zmiljkovic@mas.bg.ac.rs).

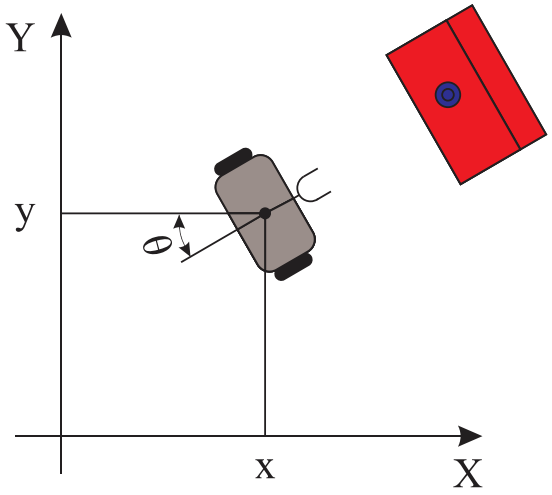


Fig. 1. Mobile robot in the environment with machine tool and part.

The related work regarding the mobile robot decision-making system is as follows. The determination of the next-best-view for environment exploring and mapping algorithm is proposed in [2]. The multi-objective decision criterion for a mobile robot with a 360° laser scanning sensor is proposed and evaluated in simulation. The proposed strategy showed superior performance compared to the other two strategies from the literature. The reinforcement learning approach for developing a mobile robot decision-making system is proposed in [3]. The mobile robot was equipped with an RGBD camera utilized to detect the obstacles. The learning approach was divided into three subtasks (i) reaching the target pose as fast as possible, (ii) obstacle avoidance, and (iii) not losing the target. According to the learned policy, a mobile robot can decide between five actions to reach the desired goal. The proposed system is verified within four simulation studies, and the results show that the proposed system achieves better results compared to the three state-of-the-art strategies.

The learning approach used for mobile robot navigation in both unknown and known environments is proposed in [4]. The model utilized for the mobile robot decision-making system is based on Developmental Networks. An incremental learning paradigm is implemented, allowing mobile robots to learn as they move in the new environment. Experimental results show that the proposed system enables mobile robots to utilize already learned cognitive functions in new environments.

A mobile robot decision-making system for outdoor path planning is proposed in [5]. The mobile robot utilizes the

 ISBN 978-86-7466-930-3

information gathered by the lidar sensor to obtain the optimal path in the uneven and obstacle-rich hill environment. In the offline stage, the robot learns in simulation the correlation of a good path with lidar data and utilizes that information in the online stage. The simulation results show the applicability of the proposed methodology for the path selection process.

Different from other approaches, in this paper, the authors propose the end-to-end trainable deep learning model capable of integrating image information and current mobile robot pose to predict the accuracy of the gripping process.

II. THE DEEP LEARNING-BASED DECISION-MAKING SYSTEM

The everlasting challenge within the deep learning-based robotic research domain is developing an adequate methodology for adapting heterogeneous data into deep learning models [6]. Examples of such data are different sensor measurements with uncertainty, a priory logical conclusions, or a mobile robot pose.

In this paper, the authors present deep learning-based decision-making system developed by modifying Resnet18 architecture [7]. The first input in the Resnet model is an image created by combining images generated by two mobile robot cameras. Images are stacked on top of each other to produce an image with the same width and height dimensions. Afterward, the additional vector input is added to the network utilized to represent the mobile robot pose (1):

$$\mathbf{x} = \begin{bmatrix} x & y & \theta \end{bmatrix}^{\mathrm{T}}. \tag{1}$$

The change in mobile robot pose is calculated with (2):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{2}$$

where v represents mobile robot translation velocity, and ω is mobile robot angular velocity.

The deep neural network architecture is presented in Fig. 2. Additional input is represented as one pixel in the image. When the feature maps are flattened just before the classification layer, the unchanged value of the mobile robot pose (from the input layer) is concatenated with the rest of the features and utilized in the classification process. The utilized Resnet18 model is created with basic and bottleneck blocks of layers with skip connections after each one; details regarding the implementation of this model can be found in [8].

Dataset for mobile robot training is generated as follows. The mobile robot is positioned to the predefined pose in the laboratory model of the manufacturing environment. Afterward, the mobile robot is set in motion until it reaches the pose close to the machine. The achieved pose is measured according to the data gathered by two wheel encoders. Then, the achieved pose is saved in the text document and two images are generated, combined, and saved. The gripping procedure is initiated, and if the mobile robot manages to grip the part, all the saved data is moved to the "successful grip" category and vice versa.

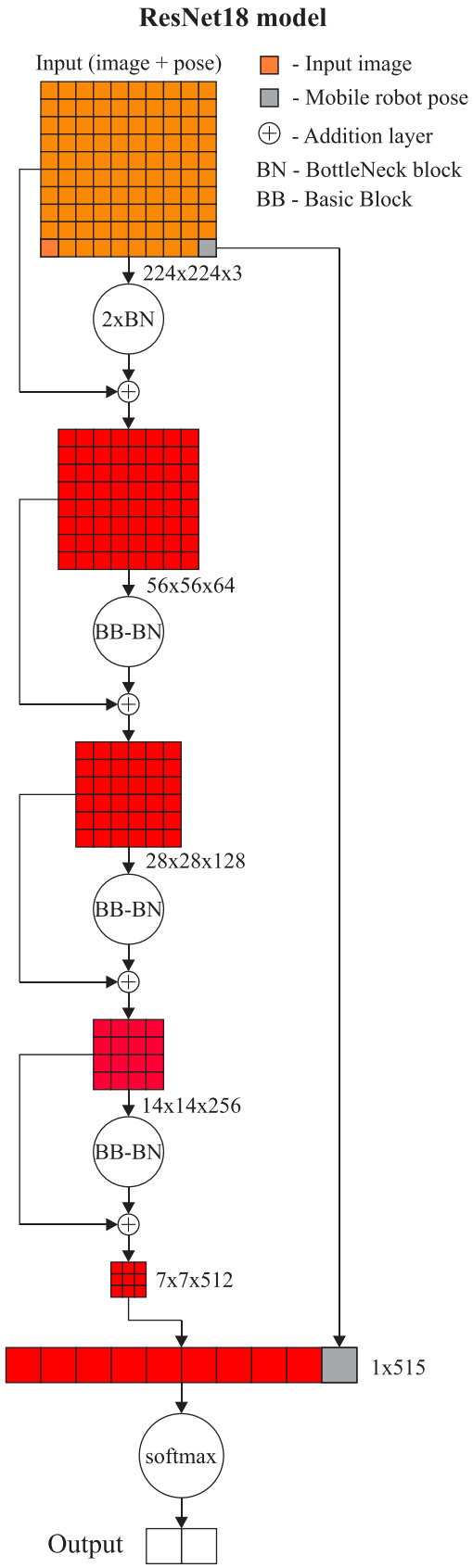


Fig. 2. Graphical representation of the proposed CNN model.

After the dataset is acquired, the training process begins. This problem belongs to the category of the binary classification process, with two outcomes "successful grip"

 ISBN 978-86-7466-930-3

and "unsuccessful grip". The classification is performed with softmax fitness function (3), and the loss function is defined with (4):

$$s_i = \frac{e^{y_i}}{\sum_{i=1}^{N} e^{y_i}} \tag{3}$$

$$\ell(\mathbf{s},\mathbf{c}) = -\sum_{i}^{N} c_i \log(s_i) \tag{4}$$

Where **y** represents the output of CNN model, *i* represents the *i*-th element of the output vector, *N* is a number of classes, **c** represents one-hot class vector.

The training is performed by stochastic gradient descent with momentum training algorithm with a batch size of one. Initial experiments are performed to determine the best training parameters for Resnet18 model. The learning rate varies from 0.0001 to 0.001, and the momentum is set to range from 0.7 to 0.9. The best performance on the training set is achieved with a learning rate of 0.0005 and momentum of 0.9.

III. Experimental results

The experimental evaluation is done by using the mobile robot RAICO (Robot with Artificial Intelligence based COgnition). RAICO is set to an initial pose in the laboratory model of the manufacturing system, and the movement command is activated. The pose RAICO achieves is relatively close to the machine where the part needs to be picked up; however, the pose is never the same due to slight differences in both the initial pose and movement process. Afterward, the final pose is calculated and integrated into the combined image generated using images from the right and left cameras. Then, the whole input is passed through the CNN network. The output represents the class (i.e., *grip* or no_*grip*) and the confidence in the class prediction. The gripping process is activated, and the outcome (successful or unsuccessful grip) is recorded. The experiment is repeated 10 times, and the results can be found in Table I.

TABLE I
THE EXPERIMENTAL RESULTS OF THE DECISION-MAKING MODEL

Exp. No.	Prediction	Confidence [%]	Successful gripping?
1	Grip	76	No
2	Grip	93	Yes
3	No_grip	83	No
4	No_grip	84	No
5	Grip	95	Yes
6	No_grip	85	No
7	Grip	85	No
8	No_grip	83	No
9	Grip	80	No
10	Grip	97	Yes

As shown in Table I, the mobile robot decision-making system adequately predicts the outcome of the gripping process in 70% of cases. Moreover, in all the cases where the prediction accuracy of the deep learning model for successful gripping is over 93%, the mobile robot actually manages to grip the part. Therefore, the high prediction confidence is highly correlated to the gripping success. Images generated by a mobile robot with prediction accuracy are shown in Fig. 3.

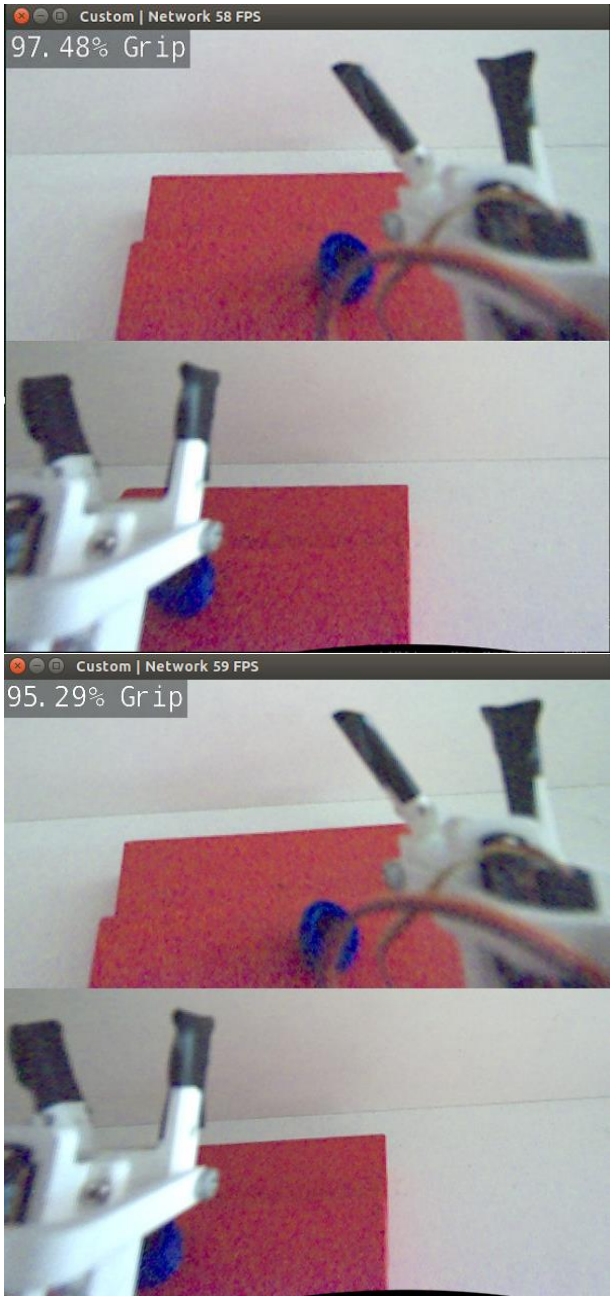


Fig. 3. Two input images for mobile robot decision-making system.

One more important piece of information that can be seen in Fig. 3 is the inference time of the developed deep learning model. Even though the proposed model has more layers and parameters than the original Resnet18, it can be used in real-time (around 60 FPS).

 ISBN 978-86-7466-930-3

Moreover, the images of the mobile robot RAICO in the laboratory model of the manufacturing system during the testing of the decision-making system are presented in Fig. 4.

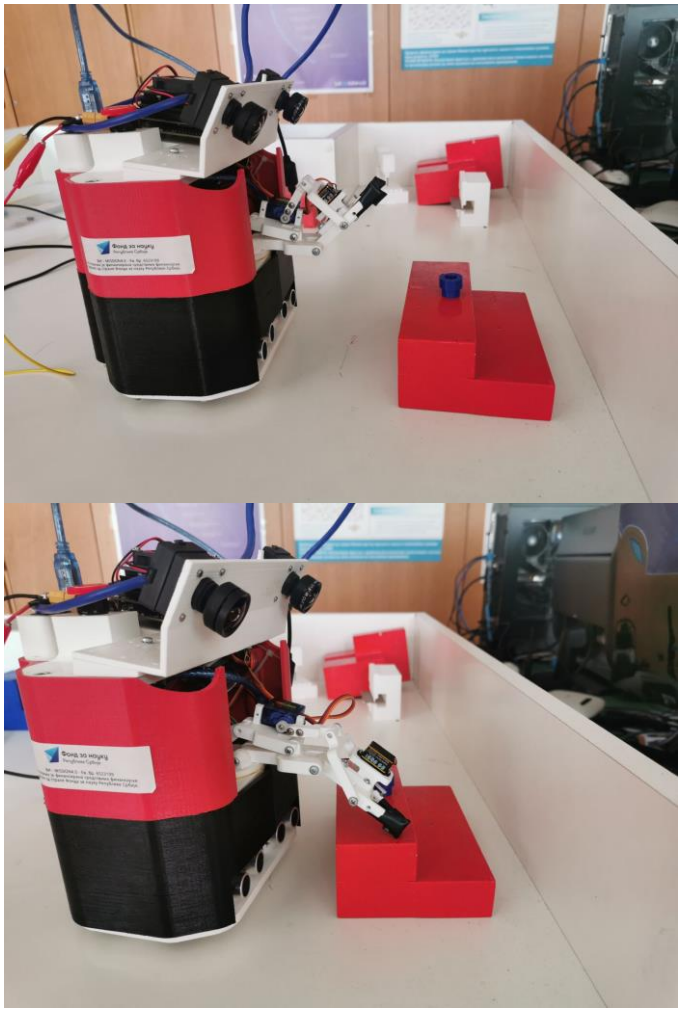


Fig. 4. Mobile robot RAICO in the pose close to the machine tool during the unsuccessful gripping process.

IV. Conclusion

In this paper, the authors propose a decision-making system based on the Resnet18 deep learning model. The input represents the images from the stereo camera pair and the pose of the mobile robot. The Resnet model is trained on the custom dataset to produce the binary classification output regarding the success of the griping process. The experimental results show that the model accurately predicts gripping success in 70% of cases. Moreover, it is experimentally verified that high confidence (93%+) in the prediction of accurate gripping has a strong correlation to the real-world successful griping process. The future research directions will include the extensive testing of the processed system with different state-of-the-art deep learning models that will enable a higher level of accuracy.

Acknowledgment

This work has been financially supported by the Ministry of Education, Science and Technological Development through the project "Integrated research in macro, micro, and nano mechanical engineering – Deep learning of intelligent manufacturing systems in production engineering" (contract No. 451-03-68/2022-14/200105), and by the Science Fund of the Republic of Serbia, grant No. 6523109, AI – MISSION 4.0, 2020 – 2022.

References

[1] Lasi, H., Fettke, P., Kemper, H. G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business and Information Systems Engineering*, *6*(4), 239–242.

[2] Basilico, N., & Amigoni, F. (2009). Exploration Strategies based on Multi-Criteria Decision Making for an Autonomous Mobile Robot. *European Conference on Mobile Robots (ECMR)*, 259–264.

[3] Hu, C., Ning, B., Xu, M., & Gu, Q. (2020). An experience aggregative reinforcement learning with multi-attribute decision-making for obstacle avoidance of wheeled mobile robot. *IEEE Access*, *8*, 108179–108190.

[4] Wang, D., Yang, K., Wang, H., & Liu, L. (2021). Behavioral Decision-Making of Mobile Robot in Unknown Environment with the Cognitive Transfer. *Journal of Intelligent & Robotic Systems*, *103*(1), 1–22.

[5] Kobayashi, Y., Kondo, M., Hiramatsu, Y., Fujii, H., & Kamiya, T. (2018). Mobile robot decision-making based on offline simulation for navigation over uneven terrain. *Journal of Robotics and Mechatronics*, *30*(4), 671–682.

[6] Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., Upcroft, B., Abbeel, P., Burgard, W., Milford, M., & Corke, P. (2018). The limits and potentials of deep learning for robotics. *International Journal of Robotics Research*, *37*(4–5), 405–420.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

[8] Jokić, A., Đokić, L., Petrović, M., & Miljković, Z. (2021). A Mobile Robot Visual Perception System based on Deep Learning Approach. *8th International Conference on Electrical, Electronics and Computing Engineering (IcETRAN 2021)*, 568–572.

 ISBN 978-86-7466-930-3

Method for Configuring Virtual Robot as an Integral Part of the Control System

Nikola Slavković, Saša Živanović, Zoran Dimić, and Nikola Vorkapić

***Abstract*—The development of integrated computing environments provides opportunities for the development of virtual production. Virtual simulation is crucial when the robot performs tasks that include some manufacturing processes. Virtual robots are used for program verification before sending it to the real robot and enable collision checking between robot segments themselves and the robot and its environment. Virtual models of industrial robots could be configured in different environments and ways. This paper presents the method for configuring virtual robots as an integral part of the control system. The virtual robot's configuration is realized under the LinuxCNC software environment and relies on OpenGL and several interface classes written in Python programming language. Developing a robot kinematic model to implement a virtual robot integrated with an open-architecture control system is necessary. Models of robot segments were imported in ASCII STL format and connected according to the robot kinematic model, and then the virtual robot was integrated within the LinuxCNC control system. The method for configuring a virtual robot as well as its kinematic model is presented in the example of the BiSCARA robot. Verifying the robot control system, virtual model, and kinematic model has been performed through several examples of drawing contours on the configured virtual robot.**

***Index Terms*—Virtual robots; Control system; Robot simulations; Kinematic modeling;**

I. Introduction

Sudden changes in production programs, product short life cycle, as well as further progress in the development of integrated computing environments for the development of new products provide opportunities to achieve the paradigm of virtual production [1]. The use of industrial robots in manufacturing and assembly lines is continuously increasing due to the need for high efficiencies, high accuracy, high production rates, and repeatability [2]. The use of robots for pick and place, welding, subtractive and additive manufacturing has been spreading in the last years with the concept of industry 4.0 [3]. In addition to performing these tasks, the application of industrial robots has been extended to laser engraving, laser and plasma cutting, robot milling, etc. Designing and testing a robotic cell is one of the essential modern manufacturing engineering tasks, especially when the robot does not just handle materials but also performs manufacturing processes [4]. To save time and verify the robot cell for performing these tasks before its production, virtual environments, i.e., virtual robots have a key role.

The notion of a virtual industrial robot is broad and includes complete models of robot structure, kinematic subsystem, tasks process, the environment, etc. All these models are integrated into a single software system, enabling some part of the virtual production.

Virtual industrial robots, that are included in the production engineering environment, could be configured in the different environments such as CAD/CAM systems, MatLab/Simulink environment, specialized CAM software for robot programming (RobotStudio, Robotmaster, …), LinuxCNC as an integral part of robot control systems, virtual reality system, etc.

This paper presents the configured virtual robots in different software environments using a five-bar, i.e., dual SCARA or BiSCARA robot. Robots based on the five-bar mechanism are widespread in academic institutions because they are suitable for experimental work. This base mechanism is almost always improved with the additional translatory and rotary axis to develop 3- or 4-axis robots. The developed virtual robot can simulate robot tasks such as laser engraving, 3-axis milling, 3D printing, etc. The first such robot, Fig. 1, is described in a US patent in 1934 [5, 6]. In 1978, Prof. Hiroshi Makino invented the well-known SCARA robot [7], while in 1985, Donald C. Fyler came up with the idea of using a five-bar mechanism as a robot [5, 8].

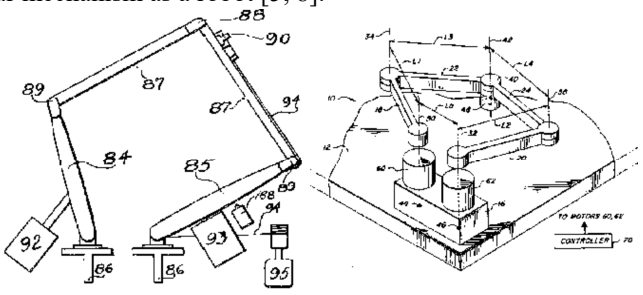


Fig. 1. Concept of five-bar, i.e., dual SCARA or BiSCARA, robot [1]

Mitsubishi Electric was the first company that commercialized the robot, MELFA RP-1A, based on this mechanism [5]. An example of a developed industrial prototype is the DexTAR robot (Dextrous Twin-Arm Robot) [9].

Associate prof. Dr. Nikola Slavković is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11120 Belgrade, Serbia (e-mail: nslavkovic@mas.bg.ac.rs).

Full prof. Dr. Saša Živanović is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11120 Belgrade, Serbia (e-mail: szivanovic@mas.bg.ac.rs).

Research Associate Dr. Zoran Dimić is with the LOLA Institute, 70A Kneza Višeslava, 11030 Belgrade, Serbia (e-mail: zoran.dimic@li.rs).

Nikola Vorkapić, MSc (ME) is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11000 Belgrade, Serbia (e-mail: nvorkapic@mas.bg.ac.rs).

 ISBN 978-86-7466-930-3

II. VIRTUAL INDUSTRIAL ROBOTS

A virtual industrial robot is a digital description of a robot, usually with simplified geometry, and is used for computer simulations of robot tasks.

To configure a virtual robot in a CAD environment [10], first, all elements of the robot structure are generated. These elements are then connected to the kinematic structure of a real robot. To be able to model a robot used in the simulation, one member of the robot mechanism must be fixed, and the moving components of the robot should be connected with appropriate kinematic connections (pin and/or slider). After this, in the next step, it is necessary to connect the coordinate systems of the workpiece and end-effector (EE) with the robot's coordinate system. In CAD/CAM environment, PTC Creo, the robot's coordinate system is defined as MACH_ZERO, while TOOL_POINT presents the end-effector's coordinate system. Coordinate systems with the same names have both an EE and a workpiece. The virtual EE is placed on the virtual robot by matching these coordinate systems. After the successful connection of the virtual workpiece and EE, a virtual industrial robot simulation according to a given program can be run. The model of the BiSCARA robot configured in the PTC Creo environment is shown in Fig. 2. The simulation of movement of the virtual robot tooltip is shown in the example of drawing a circle and a rectangle inscribed in a determined workspace of the robot.

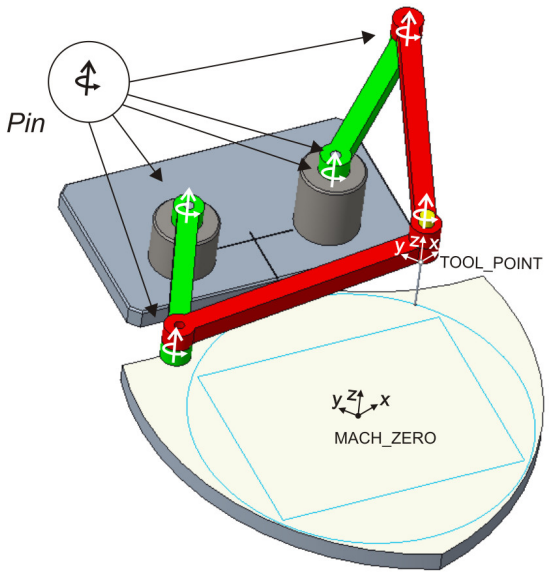


Fig. 2. Configuring of virtual robot in CAD/CAM environment

As described in the previous work [11], configuring the virtual robot in the MatLab/Simulink environment considers the development of the program for the joint space trajectory generation in the MatLab environment and configuring of the virtual robot in the Simulink environment. The structure of configured virtual robot in Simulink is presented in Fig. 3a.

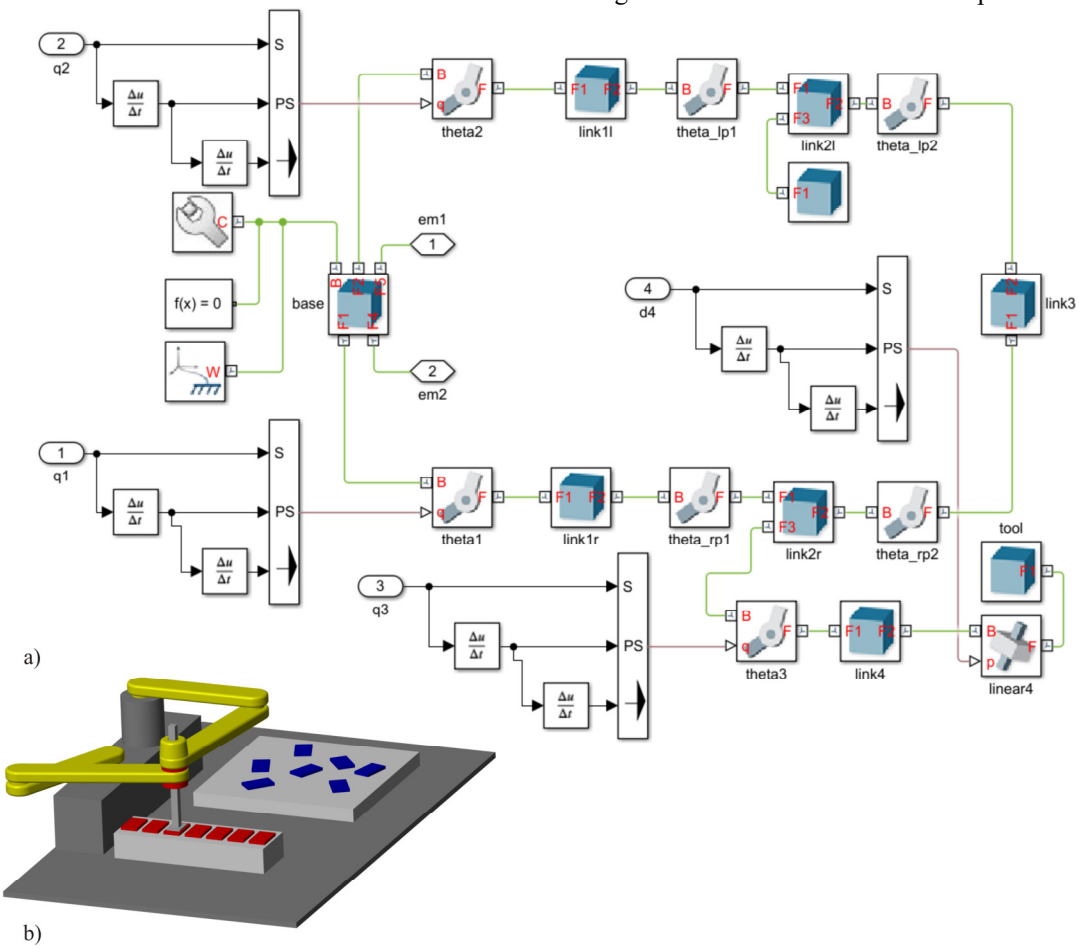


Fig. 3. Configuring of virtual robot in MatLab/Simulink environment

 ISBN 978-86-7466-930-3

It consists of a base and three kinematic chains. The base is represented with the Brick Solid element connected with World Frame, Solver Configuration, and Mechanism Configuration elements. Any kinematic chain consists of Brick Solid elements and appropriate Revolute Joint and/or Prismatic Joint elements. The active joints in these kinematics chains have to be actuated by joint time series generated in MatLab consisting of joint angle, velocity, and acceleration [11]. The model of the BiSCARA robot configured in MatLab/Simulink environment is shown in Fig. 3b.

Virtual robots, as well as virtual machines, have a significant role when developing a control system where it is necessary to integrate solutions of inverse and direct kinematics. In this way, it is possible to develop a control system even before the realized physical prototype, which is of great importance because it allows virtual testing during development, known as virtual commissioning [12, 13, 14].

This paper considers the method for configuring virtual robots in the LinuxCNC control system. Virtual robots in the control system are significant during the robot exploitation and the configuring of the control and before the realization of the robot. Program verification using virtual robots is significant because it enables:

- testing the kinematic model during the development of the control system,
- visual detection of collisions between the moving parts of the robot and between the tool, workpieces, and fixtures in the workspace,
- checking if the robot can execute the specified toolpath within the limited workspace, joints ranges, and speeds,
- training and education for robot programming, etc.

III. KINEMATIC MODEL OF ROBOT

To realize the virtual robot as an integral part of the control system, it is necessary to perform a kinematic analysis of the considered robot [11, 15]. A Parallel BiSCARA robot can be viewed as a planar manipulator with two degrees of freedom. The kinematic analysis embraced solving direct and inverse kinematic problems, determination of Jacobian matrix, and workspace analysis [11]. The kinematic model of the BiSCARA robot is shown in Fig. 4.

The robot consists of a base, a platform, and two kinematic chains with struts lengths l_1 and l_2. All mechanism elements are connected by a joint with one rotary degree of freedom. The frame {B} represents the base frame, while the platform is represented by point P because the struts of length l_2 are connected at the point P.

As it can be seen from Fig. 4, the world coordinate vector is defined as

$$ {}^{B}\mathbf{p}_P = \begin{bmatrix} x_P & y_P \end{bmatrix}^T \tag{1} $$

while joint coordinate vector is represented as

$$ \boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T \tag{2} $$

Besides another unit vector, the unit vector ${}^{B}\mathbf{a}_i$ is defined as

$$ {}^{B}\mathbf{a}_i = \begin{bmatrix} \cos(\theta_i)\cos(\gamma_i) \\ \sin(\theta_i) \end{bmatrix} \tag{3} $$

where i=1, 2 represents the number of the kinematic chain and γ_i represents the angle that defines the arrangement of kinematic chains and is introduced in order to generalize the solution of the inverse kinematic problem and parallel determination of joint coordinates during the solving an inverse kinematic problem.

Observing one kinematic chain, the following vector equations can be derived

$$ \begin{aligned} {}^{B}\mathbf{p}_p &= {}^{B}\mathbf{b}_i + k_i \cdot {}^{B}\mathbf{w}_i \\ k_i \cdot {}^{B}\mathbf{w}_i &= l_1 \cdot {}^{B}\mathbf{a}_i + l_2 \cdot {}^{B}\mathbf{z}_i \end{aligned} \tag{4} $$

from which inverse and direct kinematic problems can be solved in analytic form.

The authors' previous work represents the complete solution of inverse and direct kinematic problems, as the determination of Jacobian matrix and workspace analysis [11]. Here are presented only the solutions of the inverse kinematic problem that is crucial for developing a virtual robot integrated with a control system. The joint coordinates can be determined using the following equations

$$ \theta_i = A\tan 2(t_{i1/2}) \tag{5} $$

where

$$ t_{i1/2} = \frac{y_P \pm \sqrt{A_i^2 + y_P^2 - B_i^2}}{A_i + B_i} \tag{6} $$

and

$$ \begin{aligned} A_i &= (x_P - b_{ix})\cos(\gamma_i) \\ B_i &= \frac{l_1^2 + (x_P - b_{ix})^2 + y_P^2 - l_2^2}{2l_1} \end{aligned} \tag{7} $$

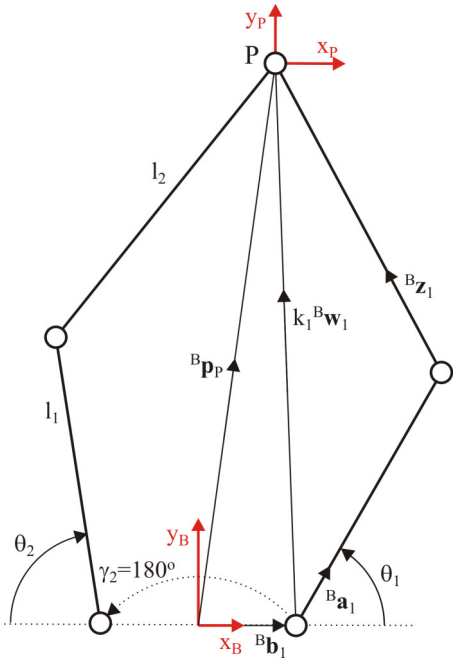


Fig. 4. Kinematic model of BiSCARA robot

 ISBN 978-86-7466-930-3

IV. METHOD FOR CONFIGURING VIRTUAL ROBOTS IN CONTROL SYSTEM

The basic concept of a configuring virtual robot in the control system is shown in Fig. 5. The virtual robot's configuration is realized under the LinuxCNC software environment and relies on OpenGL and several interface classes written in Python programming language. Python is an interpreted programming language suitable for scripting tasks, such as developing and configuring virtual environments [12].

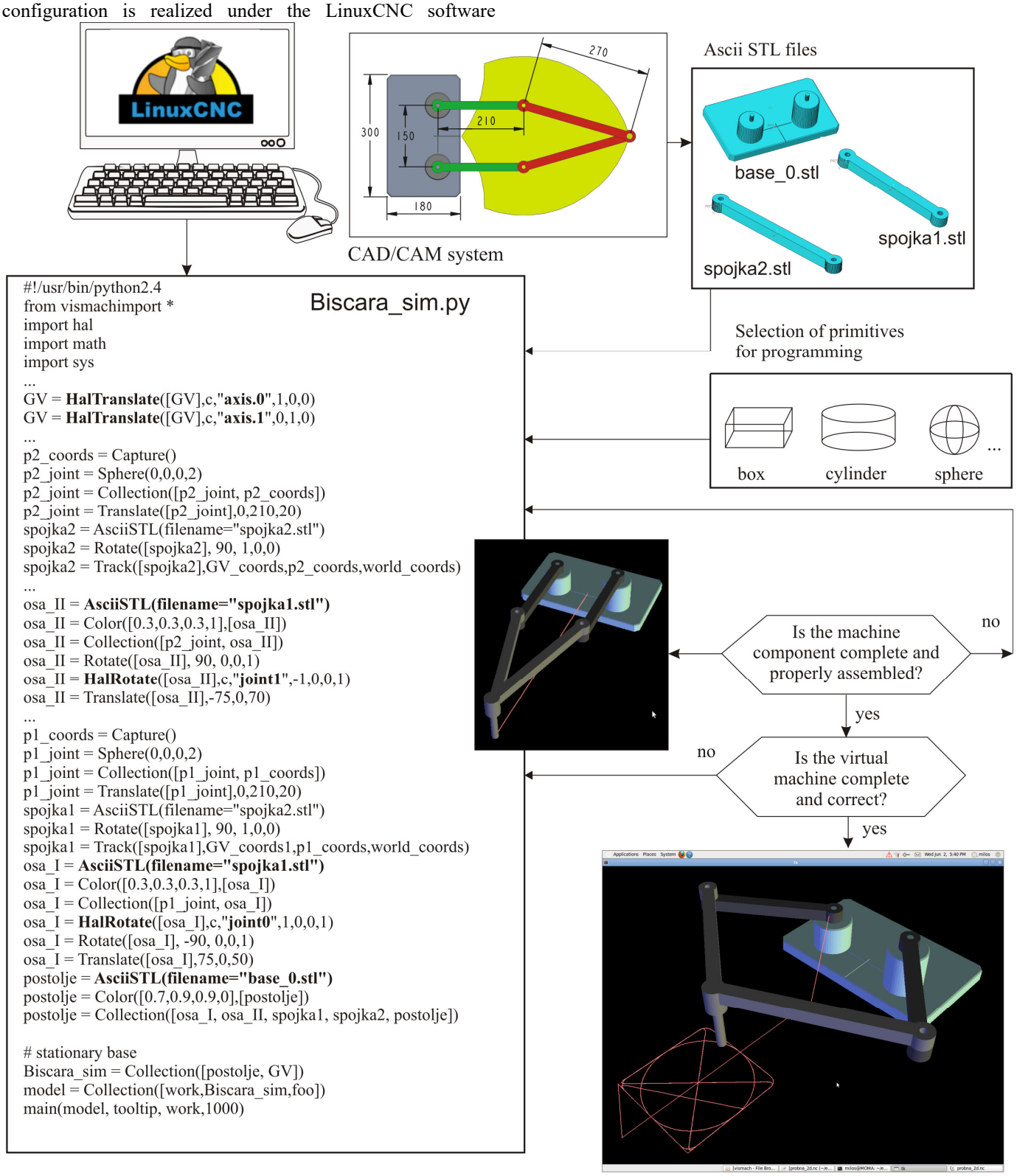


Fig. 5. The basic concept of a configuring virtual machine in a control system realized under the LinuxCNC software environment

 ISBN 978-86-7466-930-3

Models of robot components were imported in ASCII STL format and connected according to the robot kinematic model. The ASCII STL files of the robot components were obtained from the CAD/CAM system.

After the virtual robot is configured, it is integrated within the LinuxCNC control system. During the program execution for a robot in G-code, the virtual robot components are moving in real-time, according to identical control signals as for the real robot.

The modeling procedure of the virtual robot starts by defining the geometric primitives as parts of the robot assembly in CAD/CAM system. Defining only the essential and functional parts of the robot 3D model may simplify the development of the virtual robot. Basic geometric primitives, e.g., boxes, cylinders, spheres, etc., can be used to form robot parts. The positions of the geometric primitives, as well as the robot parts, are programmed according to the adopted reference frame. The robot parts are connected to form the robot base, and the moving parts of the robot, in this case, are connected via rotational joints.

Step by step programming, testing, and error correction is the only way of configuring an error-free virtual robot. Besides the manual approach, using basic geometric primitives, a CAD system can be used for preparing 3D models of robot components, which are exported as separate files ASCII OBJ or ASCII STL code. In this paper, the virtual BiSCARA robot was configured using PTC Creo. The robot components were converted into an ASCII STL format and then loaded into the Axis GUI through appropriate Python calls, Fig. 5.

Afterward, the imported components are oriented and placed in the virtual environment, resulting in a fully functional virtual robot, as shown in Fig. 5. The virtual robot is placed in a separate window and allows drawing the toolpath within the limitations of the virtual robot elements' movements.

V. Simulation and Verification

The developed virtual two-axis BiSCARA robot, integrated with the LinuxCNC control system, is suitable for drawing tool paths and laser engraving [15].

The programming system for the considered robot is entirely conventional and the same as for all machine tools which can perform milling or laser engraving. Programs for testing configured virtual robots are generated in the CAD/CAM system using trajectory milling or a special CAM system that can convert drawing in DXF format into G-code.

During the program's execution written in G-code, the virtual robot components are moving in real-time, according to the tool path, thanks to the kinematics built into the control system.

Figure 6 shows the first of the experiments on a virtual robot integrated with the control system, which represents the drawing of the programmed tool path consisting of regular geometric figures of squares with diagonals and inscribed circles.

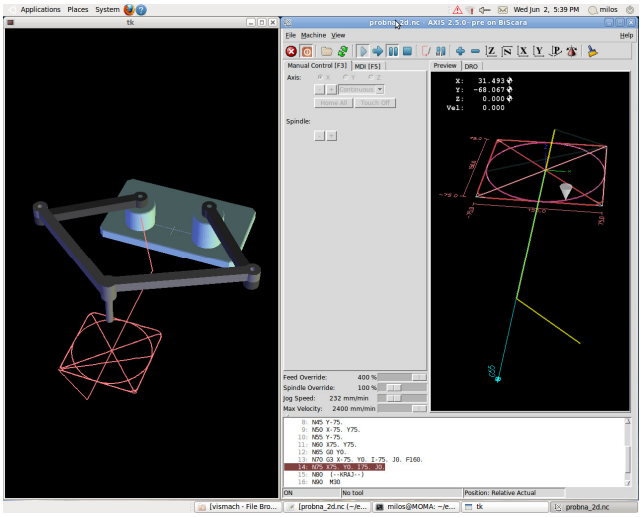


Fig. 6. Simulation of programmed tool path on virtual BiSCARA robot in LinuxCNC control system

Figure 7 shows one of the experiments on a virtual robot integrated with the control system, representing the drawing of the programmed tool path in the shape of the lion drawing. This experiment was used to test systems for programming complex 2D contours, which were programmed, based on DXF drawings and their conversion into G-code.

Some of the details during the virtual trial simulation of the BiSCARA robot are as follows: (i) the appearance of the programmed contour confirms the realized kinematic model, (ii) realized control system is correct, (iii) the used programming system gives correct G-code, (iv) positioning within the robot workspace was correct.

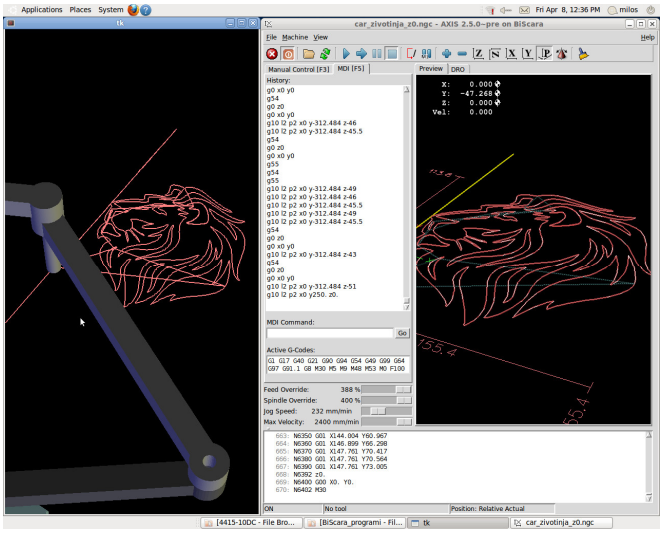


Fig. 7. Simulation of programmed tool path based on lion drawing on virtual BiSCARA robot

 ISBN 978-86-7466-930-3

VI. CONCLUSION

Testing and verifying the robotic cell before its realization through virtual environments is one of the modern manufacturing engineering tasks. Configuring a virtual environment that includes virtual robots is of crucial importance to verify the programmed trajectory, collision detection, etc.

The paper presents the configuration of a virtual robot in the CAD/CAM and MatLab/Simulink environments on the example of the BiSCARA robot. The virtual CAD and complete kinematic models are used to implement robots in the LinuxCNC software according to a presented method for configuring virtual robots as an integral part of the control system. Configuring virtual robots relies on OpenGL and several interface classes written in Python programming language. First, robot segments were imported in ASCII STL format and connected according to the robot kinematic model, and then the virtual robot was integrated.

Further research will encompass the realization of a virtual prototype of a 4-axis robot by adding on the presented base mechanism one translatory and one rotary axis. Also, the development of a laboratory prototype of a 4-axis robot will be covered in further research.

ACKNOWLEDGMENT

This research was supported by the Science Fund of the Republic of Serbia, Grant No. 6523109, AI-MISSION 4.0 as well as by the Ministry of Education, Science and Technological Development of the Serbian Government under the contract No. 451-03-68/2022-14/200105.

REFERENCES

[1] N. Vorkapić, S. Živanović, Z. Dimić, B. Kokotović, N. Slavković, "Virtual horizontal machining center LOLA HBG 80 for program verification and monitoring", *FME Transactions*, vol. 49, no. 3, pp. 696-703, 2021.

[2] J.O. Oyekan, W. Hutabarat, A. Tiwari, R. Grech, M.H. Aung, M.P. Mariani, L. Lopez, T. Ricaud, S. Singh, C. Dupuis, "The effectiveness of virtual environments in developing collaborative strategies between industrial robots and humans", *Robotics and Computer-Integrated Manufacturing*, vol. 55, pp. 41-54, 2019.

[3] W.S. Barbosa, M.M. Gioia, V.G. Natividade, R.F. Wanderley, M.R. Chaves, F.C. Gouvea, F.M. Gonçalves, "Industry 4.0: examples of the use of the robotic arm for digital manufacturing processes", *International Journal on Interactive Design and Manufacturing*, vol. 14, no. 4, pp. 1569-1575, 2020.

[4] GC Vosniakos, P. Katsaros, I. Papagiannoulis, E. Meristoudi, "Development of robotic welding stations for pressure vessels: interactive digital manufacturing approaches", *International Journal on Interactive Design and Manufacturing*, pp. 1-16, 2022.

[5] DexTAR, User's Manual, Version 1.0, by Mecademic Inc., 2014–2015.

[6] W.L.G. Pollard Jr., Spray Painting Machine, US Patent 2,213,108, filed October 29, 1934, issued August 27, 1940.

[7] H. Makino, A. Kato, Y. Yamazaki, "Research and commercialization of SCARA robot", *International Journal of Automation Technology*, vol. 1, no. 1, pp. 61–62, 2007.

[8] D.C. Fyler, Control Arm Assembly, US Patent 4,712,971, filed February 13, 1985, issued December 15, 1987.

[9] A. Joubair, M. Slamani, I.A. Bonev, "Kinematic calibration of a five-bar planar parallel robot using all working modes", *Robotics and Computer-Integrated Manufacturing*, vol. 29, pp. 15–25, 2013.

[10] N. Slavkovic, S. Zivanovic, N. Vorkapic, "Konfigurisanje virtuenog prototipa BiSCARA robota", *Časopis Tehnika-Mašinstvo*, Časopis saveza inženjera i tehničara Srbije, vol. 70, no. 3, str 311-317, 2021.

[11] N. Slavkovic, S. Zivanovic, N. Vorkapic, Z. Dimic, "Development of the Programming and Simulation System of 4-axis Robot with Hybrid Kinematic", *FME Transactions*, (in print).

[12] C.G. Lee, S.C. Park, (2014) "Survey on the virtual commissioning of manufacturing systems", *Journal of Computational Design and Engineering*, vol. 1, pp.13-222, 2014.

[13] S. Zivanovic, S. Tabakovic, M. Zeljkovic, Z. Dimic, "Modelling and analysis of machine tool with parallel–serial kinematics based on O-X glide mechanism", *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 43, no. 456, 2021.

[14] A. Rakic, S. Zivanovic, Z. Dimic, M. Knezevic, "Digital twin control of multi-axis wood CNC machining center based on LinuxCNC", *BioResources,* vol. 16, no. 1, pp.1115-1130, 2021.

[15] N. Slavković, N. Vorkapić, S. Živanović, Z. Dimić, B. Kokotović, "Virtual BiSCARA robot integrated with open-architecture control system," Proc. 14th International Scientific Conference MMA 2021 – Flexible Technologies, Novi Sad, Serbia, pp. 63-66, 2021.

 ISBN 978-86-7466-930-3

Low-cost real-time human motion capturing system

Milutin Nikolić, Lazar Milić, Milutin Studen, Mirko Raković

***Abstract*— In recent years motion capturing technology found numerous applications in industry and research areas like human-robot interaction, medical applications, etc... Those systems can be very expensive and might require a lot of setup time. In this paper, leveraging the advances in deep learning and computer science, the low-cost real-time motion capturing system is presented. The system was designed to use off-the-shelf inexpensive cameras, freely available software, and a gaming laptop. The system design, underlying math principles, reconstruction pipeline, and reconstruction results will be discussed in the paper. The presented motion capturing system can reconstruct a human pose with 33 keypoints in real-time at 17Hz. The whole setup costs less than $2500 including the price of the dedicated PC.**

***Index Terms*— Motion capturing, Human pose estimation, Human-robot interaction, Deep learning**

I. Introduction

Motion capture or "MoCap", for many years now, has been recognized in the majority of industries as the highly advanced technique of recording movements and transferring the results of such recording into digital data. It is widely used in various fields from science, through the film industry, to gaming and video game development, which makes it broadly applicable and remarkably alluring for use in different professional spheres [1].

In medicine and biomedicine, motion capture is used to conduct scientific research and analyses with regard to the understanding of human physiology, particularly the bipedal locomotion. This knowledge can contribute to understanding the effects of injuries and required rehabilitation [2]. In addition, MoCap havs the ability to record facial expressions of humans, which can help us in further understanding of human emotions. In robotics MoCap is used for constructing dynamically stable humanoid movement and for capturing motion trajectory of a human [3, 4].

Furthermore, the filmmaking and video games industry uses motion capture to record the physical actions of the live actors participating in the movies, enabling the real-life movie characters to be "transcribed" into the computer animations, i.e., digital characters. Nonetheless, the MoCap is often used for the creation of special effects in different animated content such as movies, video games, etc.[5] Also, in human-robot interaction, the MoCap is used to record, evaluate and classify human behavior patterns. Also, MoCap is extensively used to record and evaluate human behavior, and modify robot behavior based on recognized human behavior pattern [6].

Milutin Nikolić, Lazar Milić, Milutin Studen and Mirko Raković are with Faculty of Technical Sciences, University of Novi Sad. Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia (email: {milutinn, miliclazar, studen, rakovicm}@uns.ac.rs)

It may be easily concluded that, from its beginnings, motion capture has been a significant addition to the technical processes throughout the diverse industries. However, although many excellent systems leading to the outstanding results are currently present on the market, such as the Vicon Motion[1]

Capturing System, the main setback of motion capture as an asset able to simplify the variety of industrial processes is the fact that its application and practical use are conditioned upon expensive hardware and software. Furthermore, some solutions such as [7] include complex offline postprocessing following the collection of the movement data. For instance, the basic equipment, without any upgrades and advancements, that is necessary for the Vicon setup amounts to approximately $40.000, which is noticeably expensive for the SME.

Although there are many types of motion capture, optical MoCap is most commonly used technology. Optical MoCap use multi-camera setup and triangulation to determine position of markers in 3D world. There are two types of optical motion capture reflective and pulse LED. First technique has reflective markers which are placed on actor's body and because of their reflection it becomes easy for software to determine the position of markers. On the other hand, pulse LED technique has active markers that emission LED light which cameras can capture. Both techniques required 50 plus markers on actor's body for motion to be captured. This process requires a lot of time for markers to be placed on predefined part of body and also markers restrict actor's movement.

Regardless of that, the rapid development of the technology has through the years led to the possibility of the development of low-cost MoCap systems. In addition, the introduction of new methods that can extract the pose from image sequence by using artificial neural networks lowered the cost of new budget MoCap systems. Furthermore, the decrease in the prices of usable equipment for the motion capture processes, including personal computers and cameras, has affected the realization of the more affordable MoCap systems with non-equal but satisfying results [8]. Having in mind all previously mentioned, we are now able to construct budget MoCap system for less than $2500, including all hardware and software, without any markers which minimize required time for preparation, allows actor's to move freely and eliminates location restrictions. Tha low cost would enable further expansion of use of MoCap. Also, removal of required markers and long setup time, enables recording in places where we can't interfere with the subjects, e.g. recording athletes during sport events, behavioral studies of people in pubic spaces, interaction of workers with the robot

 ISBN 978-86-7466-930-3

on the factory floor, etc ... The design and implementation of such a low-cost system is the main topic of this paper.

The paper is organized as follows: in Section II we will describe the hardware components of the system and how it is layed out. In Section III the process for calibrating the system will be described. Section IV describes the whole reconstruction pipeline, while Section V gives experiment results. The paper is concluded in Section VI.

II. System setup

In this section, we will describe the complete system architecture, calibration process, and video acquisition. The video acquisition and recording system consists of three Intel RealSense D415 cameras, which are connected to a PC via a USB interface. The placement of the cameras in the room is so that each camera is oriented towards the center of the room. Since Intel RealSense D415 has narrow field of view, the coverage of the visible area is relatively small. This could potentially disrupt pose estimation. But as shown later in section results this was not a problem.

A. Intrinsic camera calibration

Because of the camera manufacturing process nature, there is a high possibility for it to be assembled imperfectly. This imperfection brings distortions to the image. There are two major distortions, and those are radial and tangential distortions.

Radial distortion has stronger effect on further points in the image and is reflected in such a way that straight lines appear curved. Similarly, tangential distortion occurs when camera lens is not perfectly aligned with image plane.

To describe radial and tangential distortion these five coefficients are used: k_1, k_2, k_3, p_1, p_2, where k_1, k_2 and k_3 are used to represent radial distortion, while p_1 and p_2 describe tangential distortion [9].

Additionally, during camera calibration, intrinsic camera parameters are calculated and these are focal length (f_x, f_y) and optical centers (c_x, c_y). Intrinsic camera and distortion parameters are calculated using OpenCV library. Process of calibration consists of multiple image captures, where each image is showing black and white chessboard with known square size. Images are then processed with the help of OpenCV functions which return all necessary parameters as the end result. Fig 1. shows example of detected chessboard and its corner points during calibration process.

B. Extrinsic camera calibration

After finding intrinsic camera parameters, it is necessary to determine camera positions in real world, also known as extrinsic camera parameters, using chessboard as a reference. Again, we are using OpenCV library to calculate these transformations and Fig **2.** shows example of images containing chessboard and its estimated location in the world.

III. Reconstruction pipeline

In order to reconstruct the pose of the human in 3D space the images coming from cameras have pass through a several processing stages. The flow chart of the raw data obtained from cameras to the final reconstructed pose is illustrated in Fig. 3. Each stage of the reconstruction pipeline will be described in detail in subsequent sections.

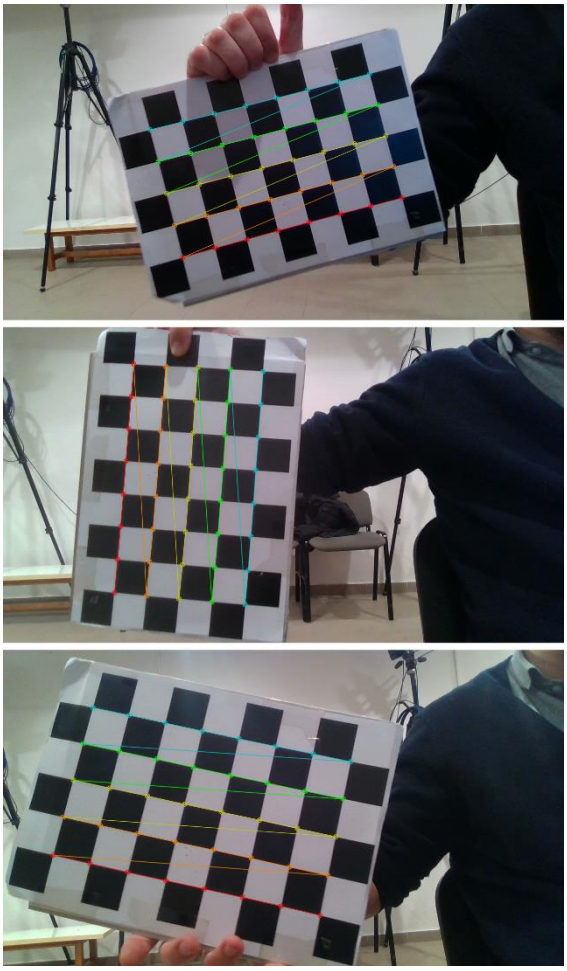


Fig 1. Images acquired for camera calibration with detected chessboards

Fig 2. Images used for extrinsic camera calibration displayed with system origin

A. Image acquisition

The first block in the processing pipeline is image acquisition from the camera. This operation is the simplest one, and the image data from the camera is retrieved by using API provided by the camera manufacturer, in this case python library `pyrealsense2` provided and maintained by Intel.

 ISBN 978-86-7466-930-3

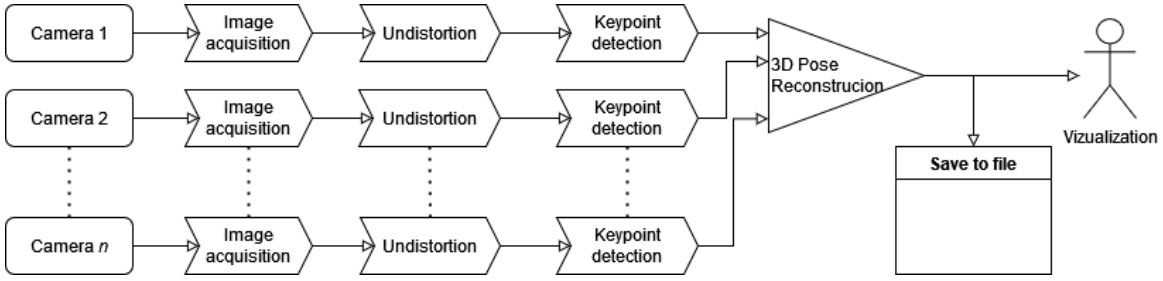


Fig 3. Processing pipeline

[10] Before retrieving the image, user has to provide camera serial number, and desired framerate.

B. Undistortion

The obtained image contains some distortion due to lens imperfection and assembly errors. Such a distortion will introduce errors in the reconstructed pose, so it is extremely important to correct it. In section III we have described the camera calibration process, and result of it is the distortion parameters. In this block, those distortion parameters will be used to undistort the obtained image.

C. Human pose detection

After obtaining distortion-free images the next step is the detection of the human keypoints on the image. Until very recently, without special clothing or markers, accurate detection of body keypoints was almost impossible. In year 2017 OpenPose was developed as a first convolutional neural network based human pose detection framework.[11] It was trained using COCO dataset and it can detect 25 keypoints on the human body. Also, it can detect multiple humans captured by the camera.

Unfortunately, OpenPose, although very accurate, is very computationally demanding. In order to run OpenPose real time, dedicated GPU is required which would significantly increase the total system cost.

Alternative was found in MediaPipe [12] developed by Google as an open-source cross platform, customizable machine learning solution set for live and streaming media. One of solutions provided is used for human pose detection and tracking. It can infer 33 3D keypoints and background segmentation mask on the whole body from RGB video frames utilizing BlazePose [13]. The detected keypoints are given in Fig 4. The MediaPipe can be set up to detect even higher fidelity mode, with total 543 keypoints, with 33 keypoints on the body, 21 on each hand and 468 on the face. To be able to reconstruct face and whole body, the higher resolution and quality camera is required, and for that reason we decided to reconstruct only 33 body keypoints. One limitation of this framework compared to OpenPose is that it can detect only one human in the scene. Multiple humans would inevitably produce detection errors.

3D poses are given relative to pelvis, whose 3D pose is unknown, and, if just one camera is used, can suffer from unknown scale, leading to wrong 3D pose estimation. Hence, we still need multi-camera setup to obtain full 3D position of the keypoints.

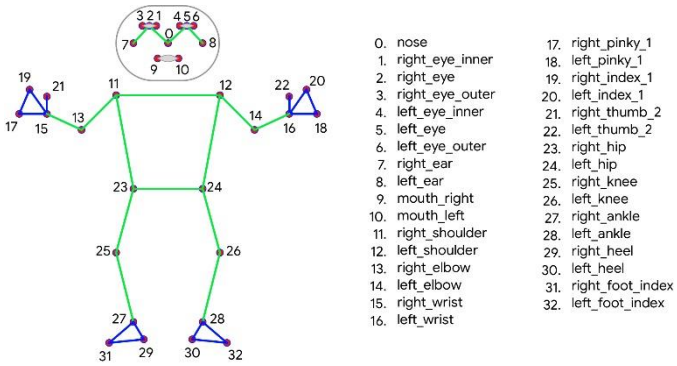


Fig 4. Keypoints detected by BlazePose

D. 3D reconstruction

After getting positions of all keypoints on all cameras, the last step is to reconstruct their positions in a 3D space. To do so, let's assume that we have *n* cameras. The keypoint in space **Q** can be seen on camera *i* at the location $\mathbf{q}_i$ given in pixels. The position and rotation matrix of camera *i* are given with $\mathbf{T}_i$ and $\mathbf{R}_i$. The intrinsic camera matrix is $\mathbf{M}_i$. These vectors and matrices are the result of the calibration process described in section IIIA. The setup is illustrated in Fig. 5. Now, having all the camera positions, rotation matrices and location of the keypoint on all images, the goal is to compute the position of the point **Q** in the 3D space.

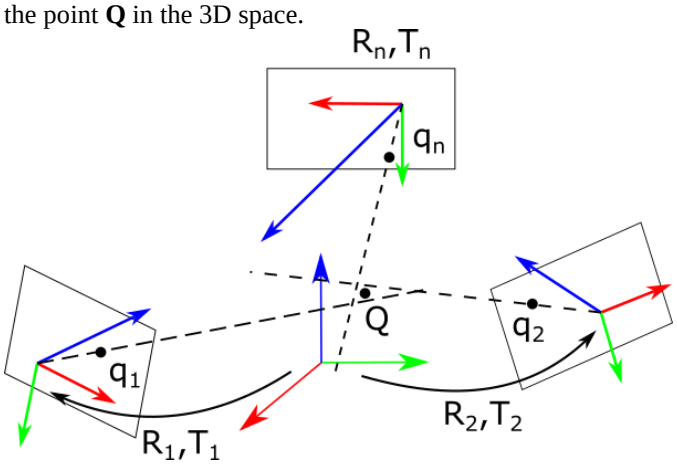


Fig. 5 Setup with multiple cameras observing the same point.

To derive equations, we will start from pinhole camera model, and projection rule:

$$\mathbf{q}_i = s_i \mathbf{M}_i \mathbf{R}_i^{\mathrm{T}} (\mathbf{Q} - \mathbf{T}_i) . \quad (1)$$

 ISBN 978-86-7466-930-3

Where s_i is positive scalar and $\mathbf{M}_i$ is intrinsic camera matrix projecting points from the space to pixels in image. From that equation we can deduce $\mathbf{Q}$:

$$\mathbf{Q} = \mathbf{T}_i + w_i \mathbf{R}_i \mathbf{M}_i^{-1} \mathbf{q}_i = \mathbf{T}_i + w_i \mathbf{u}_i, \quad (2)$$

where we w_i is again positive scalar equal to $1/s_i$. The physical meaning is that a point $\mathbf{Q}$ lies on the ray starting from center of camera i $\mathbf{T}_i$, that passes through point $\mathbf{q}_i$ and follows direction $\mathbf{u}_i$.

Obviously, this is the ideal case, where cameras are perfectly calibrated and keypoint is perfectly detected. In the real world we can't achieve that, so the mentioned ray will pass close, but will not contain the point $\mathbf{Q}$. Including that observation we can rewrite equation (2):

$$\mathbf{Q} = \mathbf{T}_i + w_i \mathbf{u}_i + \boldsymbol{\varepsilon}_i, \quad (3)$$

where $\boldsymbol{\varepsilon}_i$ represents the deviation of point $\mathbf{Q}$ from ray going from $\mathbf{T}_i$, in a direction $\mathbf{u}_i$.

The position of the point Q will be obtained by minimizing the deviation from ideal rays. It is the result of the following optimization problem:

$$\min_{\mathbf{Q}} \sum_{i=1}^{n} \left\| \boldsymbol{\varepsilon}_i \right\|^2. \quad (4)$$

The meaning of this optimization problem is that the point $\mathbf{Q}$ that is the detected on n images at location $\mathbf{q}_i$ is the point in space whose total squared distance to all rays starting at $\mathbf{T}_i$, and passing through point $\mathbf{q}_i$ is minimal.

By expressing inserting $\boldsymbol{\varepsilon}_i$ from equation (3) and inserting it into (4) and some matrix manipulation we can the optimization problem as:

$$\min_{\mathbf{Q},\, w_i} \left\| \begin{bmatrix} 1_{3x3} & -\mathbf{u}_1 & 0_{3x1} & \dots & 0_{3x1} \\ 1_{3x3} & 0_{3x1} & -\mathbf{u}_2 & \dots & 0_{3x1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1_{3x3} & 0_{3x1} & 0_{3x1} & \dots & -\mathbf{u}_n \end{bmatrix} \begin{bmatrix} \mathbf{Q} \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} - \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_3 \end{bmatrix} \right\|^2 \quad (5)$$

Such an optimization problem has a well known solution:

$$\begin{bmatrix} \mathbf{Q} \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} 1_{3x3} & -\mathbf{u}_1 & 0_{3x1} & \dots & 0_{3x1} \\ 1_{3x3} & 0_{3x1} & -\mathbf{u}_2 & \dots & 0_{3x1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1_{3x3} & 0_{3x1} & 0_{3x1} & \dots & -\mathbf{u}_n \end{bmatrix}^{+} \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \end{bmatrix} = \mathbf{A}^{+} \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \end{bmatrix} \quad (6)$$

where we introduced substitute $\mathbf{A}$ for shorter notation and plus (+) represents a MP-inverse matrix.

It is important to note that matrix $\mathbf{A}$ has 3n rows and 3+n columns and for it to be tall, guaranteeing unique solution n has to be at least 2, meaning that we need to see the waypoint at, at least two cameras to be able to reconstruct the pose. Detecting the same waypoint on more than 2 cameras will produce more accurate result. Also, matrix $\mathbf{A}$ is different for each keypoint, so in order to reconstruct model with e.g 33 keypoints, we need to construct 33 different matrices and solve 33 different optimization problems.This process has to be performed for each recorded frame.

E. Data post processing

At this point in pipeline, we have already fully reconstructed the 3D pose of all the keypoints. The last basically depends on the usage of MoCap system. We can either save data for later use, or directly stream it to dedicated video/gaming production software, medical application, etc …

IV. RESULTS

Complete process of image acquisition, undistortion, pose estimation and 3D reconstruction is done on laptop with Intel Core i7 10750H, 6 physical & 12 logical core 10th generation CPU with 16GB of RAM and Nvidia GeForce 1660TI GPU. Even so, we were able to achieve stable 20 FPS. The code has been implemented on Ubuntu 20.04 Focal Fossa in Python3 with extensive use of `multiprocess` library, enabling parallelization. Separate process responsible for image acquisition, undistortion and pose estimation was spawned for each camera, with one additional process tasked with 3D reconstruction, and data post processing.

Fig 6. A frame from all 3 cameras taken at the same time with detected keypoints drawn over

 ISBN 978-86-7466-930-3

In Fig 6, the images from all three cameras taken at the same time are shown. The keypoints detected by MediaPipe are overlaid on top of the undistorted images. I can be noted that not all markers are detected on all 3 images, e.g right arm is not detected on first image. Nevertheless, as seen in Fig 7 it can be observed that the system was able to reconstruct the full human pose, with all keypoints. That comes from the fact that for 3D reconstruction we need a keypoint to be detected at minimum 2 cameras. Detection of a keypoint is more than 2 cameras, increases estimation accuracy. It can be noted that the reconstructed full body pose closely matches the one shown in undistorted images.

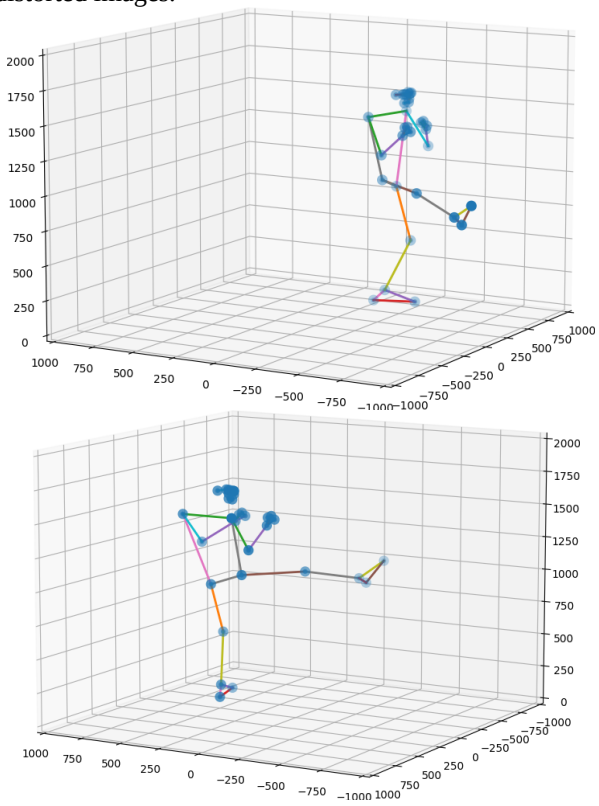


Fig 7. Full body pose reconstructed from images shown in Fig.6

However, during pose estimation some problems occurred. We can observe several frames with incorrectly detected human poses as shown in Fig. 8. Those bad reconstructions can be a result of several issues. Firstly some body parts can go out

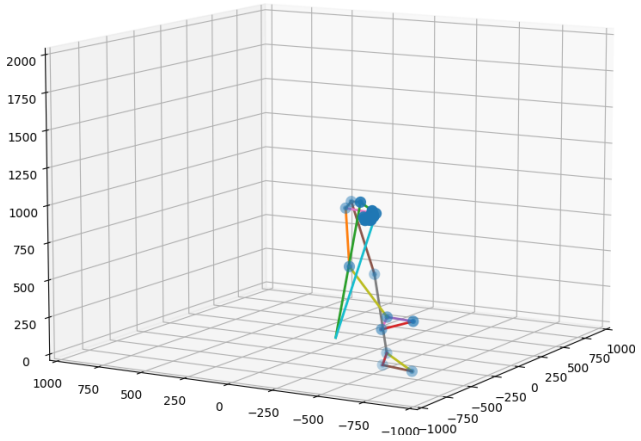


Fig 8. Incorrectly reconstructed human pose

of recording volume or be occluded by objects in the scene, thus preventing MedaPipe to detect keypoints successfully. Other issues can come out of just wrong detection pose detection as shown in the middle of Fig 9. These images show MediaPipes' inability to estimate human pose in non-standing positions. That is a result of MediaPipe being trained on dataset which consists mostly of humans in standing position. That introduces strong bias in the CNN, which expects the torso to be upright. In Fig 9. that was not the case, and MediaPipe has detected a chair in the background as humans face and shoulders. That frame prevents successful reconstruction.

V. Conclusion

In this paper we have presented a budget MoCap system. The system uses inexpensive of-the shelf cameras with a middle-range gaming laptop. Used libraries and software are all open source, with licenses allowing commercial use. Hence, most of the price of the system comes from gaming laptop. The price of the full system was under $2500, but uses common hardware that we had at our disposal, so the MoCap didn't cost us any money. The framerate that we could have achieved was 20FPS. The reconstructed poses closely matched the pose of the actor that was recorded and were stable as long as the actor was within recording volume.

Except low price, the system doesn't require any markers, which significantly reduces setup time. There is no need for

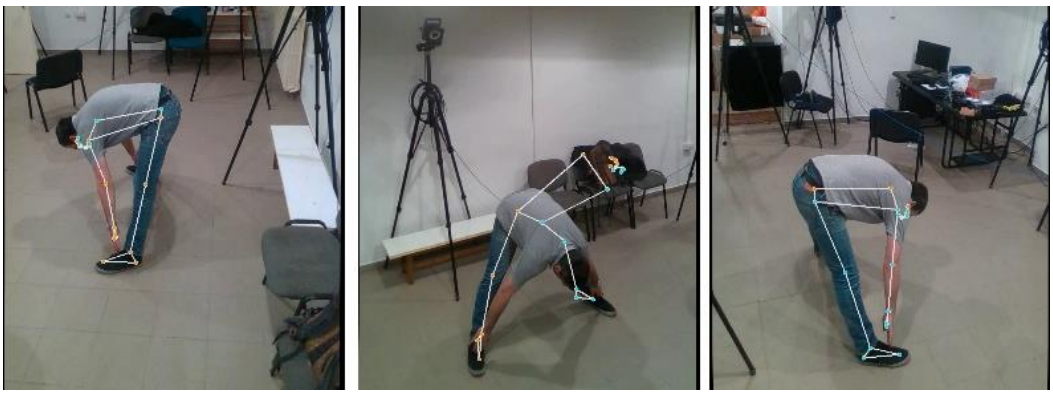


Fig 9. A frame from all 3 cameras taken at the same time with detected keypoints drawn over with incorrect human pose detections

 ISBN 978-86-7466-930-3

tedious process of attaching markers to the actor. The other benefits from this is that people can be recorded while performing their task without interference. That means, the person's motion isn't modified by the fact that there are markers attached to it's body. For behavioral studies, person's motion can be recorded subject's knowledge, reducing the chance for the person to unconsciously alter the behavior. This makes developed MoCap, extremely potent for studies in sport science and behavioral studies with potential HMI application.

Although presented MoCap has a lot of potential there are a few issues that would be the topic for the future research. The current approach doesn't include the temporal continuity of the recording, but each frame is considered independently. The system doesn't consider human model, which would make detection more accurate and more robust to false detections and outliers. Finally, current deep learning backend, MediaPipe can detect only one person. OpenPose, can be used instead, but that comes at the cost of framerate and ability to run system in real time or at the price of higher performance and higher price hardware.

REFERENCES

[1] Menolotto, M., Komaris, D. S., Tedesco, S., O'Flynn, B., & Walsh, M. (2020). Motion capture technology in industrial applications: A systematic review. Sensors, 20(19), 5687.

[2] Schönauer, C., Pintaric, T., Kaufmann, H., Jansen-Kosterink, S., & Vollenbroek-Hutten, M. (2011, June). Chronic pain rehabilitation with a serious game using multimodal input. In 2011 International Conference on Virtual Rehabilitation (pp. 1-8). IEEE.

[3] Ogrinc, M., Gams, A., Petrič, T., Sugimoto, N., Ude, A., & Morimoto, J. (2013, May). Motion capture and reinforcement learning of dynamically stable humanoid movement primitives. In 2013 IEEE International Conference on Robotics and Automation (pp. 5284-5290). IEEE.

[4] Ito, T., Ayusawa, K., Yoshida, E., & Kobayashi, H. (2020). Simultaneous Control Framework for Humanoid Tracking Human Movement with Interacting Wearable Assistive Device. IEEE Robotics and Automation Letters, 5(2), 3604-3611.

[5] Duarte, N. F., Raković, M., Tasevski, J., Coco, M. I., Billard, A., & Santos-Victor, J. (2018). Action anticipation: Reading the intentions of humans and robots. *IEEE Robotics and Automation Letters*, *3*(4), 4132-4139.

[6] Sharma, S., Verma, S., Kumar, M., & Sharma, L. (2019, February). Use of motion capture in 3D animation: motion capture systems, challenges, and recent trends. In 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon) (pp. 289-294). IEEE.

[7] Ikegami, Y., Nikolić, M., Yamada, A., Zhang, L., Ooke, N., & Nakamura, Y. (2020). Whole-Game Motion Capturing of Team Sports: System Architecture and Integrated Calibration. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 10256-10261). IEEE.

[8] Thewlis, D., Bishop, C., Daniell, N., & Paul, G. (2013). Next-generation low-cost motion capture systems can provide comparable spatial accuracy to high-end systems. Journal of applied biomechanics, 29(1), 112-117.

[9] Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.".

[10] Intel RealSense SDK2 https://dev.intelrealsense.com/docs/python2 Accessed: May 8. 2022.

[11] Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7291-7299).

[12] MediaPipe Pose Module, https://google.github.io/mediapipe/solutions/pose, Accessed May 8, 2022

[13] Bazarevsky, V., & Grishchenko, I. (2020). On-device, real-time body pose tracking with mediapipe blazepose. *Google AI Blog*.

 ISBN 978-86-7466-930-3

GAN-based Data Augmentation in the Design of Cyber-attack Detection Methods

Dušan Nedeljković and Živana Jakovljević, *Member, IEEE*

***Abstract*—The advent of the Industry 4.0 paradigm that relies on the concepts of Cyber-Physical Systems (CPS) and the Industrial Internet of Things (IIoT) leads to the transition from centralized to distributed control. In this approach, interconnected smart devices (sensors, actuators, etc.) as the key enablers achieve system control through coordinated work. Introduction of IIoT leads to ubiquitous communication between smart devices, thus opening up a vast area for potential malicious threats and attacks which can cause serious consequences, take to system dysfunction or even endanger human lives. Therefore, security mechanisms have to be developed to provide timely detection of different cyber-attacks and to keep the system safe and protected. Since industrial processes are often very complex and their analytical model is very difficult to determine, deep learning based methods for cyber-security mechanisms development are imposed as a technique of choice. Successful employment of data-driven solutions, particularly based on deep learning approaches usually requires a big amount of data. However, due to various limitations in the acquisition of data from the real process, its availability is still a major challenge. For instance, the Industry 4.0 factory implies frequent reconfiguration which reduces the time intervals available for experimental procedures such as data acquisition. One of the ways to deal with this issue is called data augmentation. In this paper, we apply data augmentation in the design of cyber-attack detection methods in Industrial Control Systems (ICS). In particular, we explore the possibilities for utilization of Generative Adversarial Networks (GAN) to generate the necessary amount of data for deep learning based modeling using a relatively small number of available samples on input.**

***Index Terms*—Data augmentation; Cyber security; Generative Adversarial Networks; Deep learning; Convolutional Neural Networks.**

I. Introduction

IMPLEMENTATION of Cyber-Physical Systems (CPS)-based smart devices at industrial plants represents the basis for the digitization of manufacturing processes and leads to the next step in industrial evolution known as Industry 4.0 [1]. Industrial Control Systems (ICS) embrace Industrial Internet of Things (IIoT) and go through significant changes. In addition to enormous benefits, introduction of IIoT at shop-floor has a number of drawbacks, where the ubiquitous wired or wireless communication between smart devices and connection of ICS to Internet can be singled out. Since ICS are no longer isolated, communication links between IIoT devices become vulnerable for the attacks by different malicious adversaries (Fig. 1). To address this issue specially designed systems for ICS cyber-attacks detection and communication links protection are necessary.

Considering the real-time operation of ICS and safety related issues such as catastrophic damages or even threat to human lives that cyber-attacks can bring about, the requirements for cyber-security systems in ICS differ from general information technologies (IT). Opposite to general IT where the data confidentiality is paramount, in ICS data availability, along with its integrity, is the most important [2, 3]. Another key aspect of ICS is that expected lifetime of ICS components (at least 10-15 years) is significantly longer than in general IT (3-5 years) [4] and that, as a result, ICS contain a large number of devices that use legacy communication protocols that do not even utilize the basic protection mechanisms such as authentication [5]. In addition to multilayered protection mechanisms based on network segmentation and segregation, one of the key concepts for cyber-security in ICS is Defense-in-Depth. This concept implies the ability of the devices within ICS to recognize an attack if it bypasses previous layers and that the device should do this before the attack achieves the desired effects [6]. For these purposes host based (installed on the device) Intrusion Detection Systems (IDS) are developed.

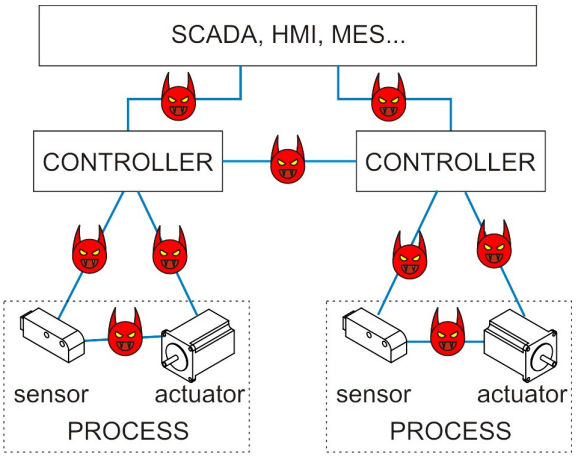


Fig. 1. Cyber-attacks on communication links in ICS within Industry 4.0

Generally, IDS within ICS are based on the model of the system behavior or the data communicated between system

Dušan Nedeljković, MSc (ME) is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11000 Belgrade, Serbia (e-mail: dnedeljkovic@mas.bg.ac.rs).

Prof. Dr. Živana Jakovljević is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11000 Belgrade, Serbia (e-mail: zjakovljevic@mas.bg.ac.rs).

 ISBN 978-86-7466-930-3

elements, and the attack is detected as the discrepancy between modeled and exhibited behavior of the system or as the discrepancy between modeled and data received through communication links. Depending on the approach for the model generation, IDS within ICS can be classified in two high level categories: design based and data based. In the design based approaches the model of the system/data is obtained in a mathematically formalized way using analytical models or different formal methods depending on the system type [7, 8].

Data based approaches, on the other hand, use the data obtained during system operation to create the model usually using different machine learning (ML) techniques. These approaches can be supervised, semi-supervised and unsupervised [9]. Supervised methods use labeled datasets containing data obtained during normal system operation and during system operation under attacks to generate detection mechanisms [10, 11]. Unsupervised methods, on the other hand, generate IDS using unlabeled data again containing the data obtained from the system performing with and without attack, and ML techniques find the structure within data themselves. The main shortcoming of the considered classes of methods is that they show low generalization properties when attacks not present during IDS creation are exhibited on the system.

Finally, semi-supervised methods use only the data obtained during normal system operation, i.e., from the system that was not subject to the attacks. They generate the model of the system behavior/communicated data during normal operation and the attack is recognized as the discrepancy between exhibited and the values estimated using the developed models. This class of approaches is most commonly utilized and shows better generalization to different kinds of attacks [12, 13, 14].

In our previous work [14] we have proposed a method for the development of IDS on communication links within ICS that is based on Convolutional Neural Networks (CNN). The proposed method, as well as the other data driven methods, requires significant amount of data from the real-world ICS that cannot be always easily obtained. In this paper we explore the possibility of augmentation of real-world data using Generative Adversarial Networks (GAN) and utilization of thus obtained dataset for IDS creation.

The reminder of the paper is structured as follows. Section 2 briefly describes GAN, whereas Section 3 refers to the utilized method for the development of IDS in ICS. Performance of IDS created using the data generated with a GAN and its comparison with IDS created based on real-world data are shown in Section 4 using an example of electro-pneumatic positioning system based on smart devices. Finally, in Section 5 we provide conclusions and future work guidelines.

II. Generative Adversarial Networks

Generative Adversarial Networks (GAN) represent a method for creation of generative model using adversarial process [15]. GAN consists of two players:

- Generator G that has the goal to generate data with the distribution close to the distribution of training data (Fig. 2.a), and
- Discriminator D with the goal to recognize if the data is created by generator or comes from the original dataset through classification of input as real or generated (fake) data (Fig. 2.b).

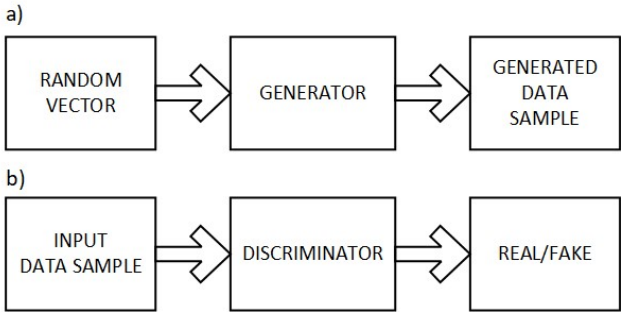


Fig. 2. Generative Adversarial Network: a) Generator, b) Discriminator

GAN training is a two player adversarial game in which generator generates fake data and tries to force discriminator to make a mistake and to recognize this data as real [16, 17]. As a rule, at the generator input is a vector of random numbers (vector of latent variables) and at its output is multidimensional vector that represents the generated data (Fig. 2.a). This data is at the input into discriminator, whereas at the discriminator output is a scalar which represents the probability that the input data is real and classifies the input accordingly (Fig. 2.b).

Generator and discriminator are trained simultaneously, where generator creates a batch of fake samples that are along with a batch of real samples from training dataset put to discriminator to classify them [16]. Based on the quality of discriminator's classification, the generator is updated to create "better" fake data and discriminator is updated to perform better classification. This adversary game repeats for a predefined number of iterations.

Generator and discriminator can be in the form of different ML based models. In our approach we will use deep neural networks (DNN), in particular CNNs and fully connected neural networks – Multilayer Perceptron (MLP).

III. CNN-based Method for the Development of IDS in ICS

The CNN-based method for the development of IDS in ICS that we have proposed in our previous work [14] belongs to the class of semi-supervised data driven methods and consists of the offline and online phase. During offline phase it generates the CNN based model of signals transmitted between IIoT devices using the data acquired during normal system operation. The model is based on auto-regression of the transmitted signals where the current value of the signal is estimated using a buffer of previously received v values. The

 ISBN 978-86-7466-930-3

main characteristic of this method is that it is designed to autonomously find the CNN with relatively small number of parameters that models the training data with good accuracy, opposite to alternative approaches that are as a rule based on manual trial and error.

IDS offline development consists of three main steps (Fig. 3). The first step represents signal preprocessing that performs FIR filtering, data structuring in ordered pairs prepared for training and data shuffling. CNN hyper-parameters (number of CNN layers, size and number of filters within them, number of pooling layers and their parameters, number of dense layers and number of neurons within them, etc.) are varied in the second step, and for each combination a CNN model is created. Finally, in the third step the generated model is selected as appropriate if it satisfies the following criteria:

1. The variance between real and estimated values should be similar for test and training data; this insures that the model is not prone to overfitting or underfitting.
2. The simulation of online performance of the IDS based on the developed model should show good performance in terms of false positive attacks detection; this insures the robustness of IDS to false attack detection online which is very important for smooth operation of ICS.

Once the model that meets given criteria is encountered, the offline procedure stops and this model is put to online detection of attacks. During online phase, the attack is detected based on the discrepancy between estimated and signal values received through communication links. If this discrepancy is higher than threshold automatically calculated from training data for z consecutive signal samples, the attack is detected.

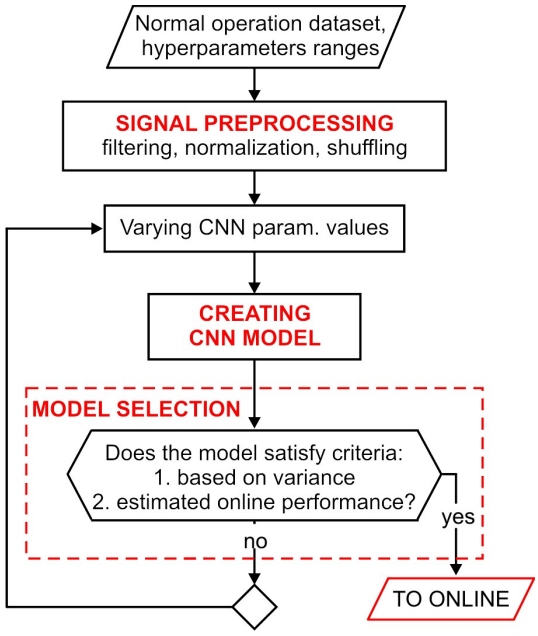


Fig. 3. Overview of the method for IDS development – offline phase

As can be observed from the presented short overview of the method, as all DNN based methods, it is highly dependent on the quality and quantity of the training data and it requires a large amount of this data at input. Acquisition of the data from the system in normal operation requires that the system is operated in isolated conditions without possibility for the attacks. Here the question arises if it is possible to operate the system long enough in such conditions to get sufficient amount of data. Another important issue that is present in Industry 4.0 factory is frequent reconfiguration of resources which leaves little room for experimenting with it. So the acquisition of appropriate amount of data from the system can be hardly feasible in some situations. For this reasons in the following section we explore the possibilities to use relatively small amount of data from real process and to augment it with data obtained using GAN.

IV. The Development of IDS for Electro-Pneumatic Positioning System based on Data Generated using GAN

In this paper we will develop IDS using GAN generated data for experimental electro-pneumatic positioning system – DisEPP (Fig. 4) that consists of:

1. Smart pneumatic cylinder, based on rodless cylinder driven by electro-pneumatic pressure regulator (EPR) that regulates pressure in 2-6 bar range on one and by mechanically controlled pressure regulator (MR) set to 4 bar on the other side, that is controlled by local controller LC1;
2. Smart encoder based on magnetic linear encoder controlled by LC2.

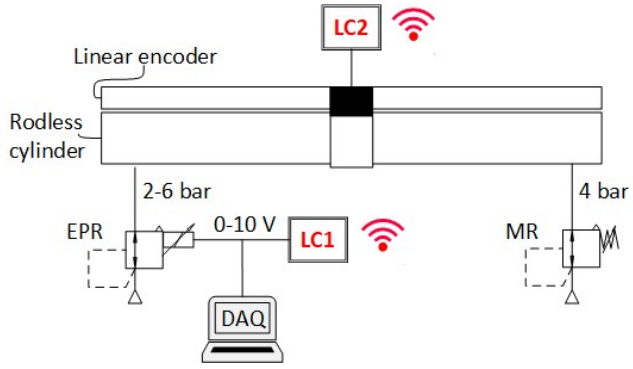


Fig. 4. A schematic representation of electro-pneumatic positioning system DisEPP

Both, LC1 and LC2 represent “mbed” devices based on ARM Cortex-M3 running at 96 MHz [18] augmented by IEEE 802.15.4-compliant wireless transceiver Microchip MRF24J40MA [19] that is used for communication between devices. The control task given in the form of desired piston positions is distributed between LC1 and LC2 in such way that: (i) LC2 has desired trajectory at input and calculates the corresponding pressure on electro-pneumatic regulator using PID and sensory signal; (ii) LC2 communicates PID output to

 ISBN 978-86-7466-930-3

LC1 using IEEE 802.15.4; (iii) LC1 converts received PID output to the 0-10 V range and puts it to electro-pneumatic pressure regulator which finally sets the pressure to 2-6 bars (proportional to the received voltage) and invokes the piston movement.

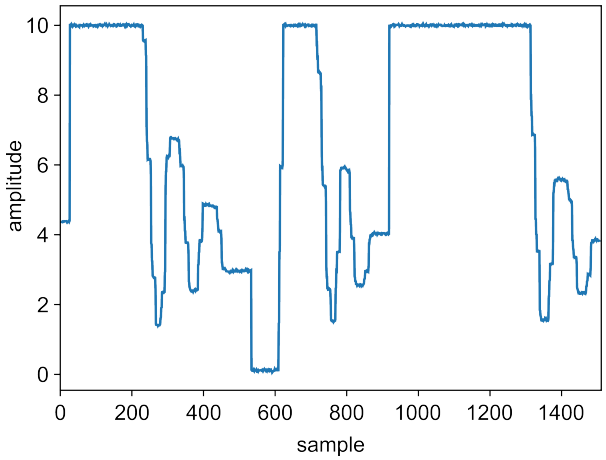


Fig. 5. An excerpt of signal acquired from DisEPP

IEEE 802.15.4 communication link between LC1 and LC2 represents vulnerable point for cyber-attacks and should be protected using IDS. The first step in IDS design using method from section III is generation of dataset representing signals obtained during normal system operation. For this purpose, we have acquired the voltage put to electro-pneumatic pressure regulator using National Instruments Data Acquisition (DAQ) system operating with 100 Hz sampling rate. A total of 399,000 samples x_i, $i \in [1, 399{,}000]$ were acquired and an excerpt of 1,500 samples is presented in Fig. 5.

A. Data Augmentation using GAN

To augment the data acquired from DisEPP we have developed a GAN with the following elements. The discriminator (Fig. 6) has the following architecture:

1. Two blocks of: (i) convolutional layer with 30 filters, each containing 10 samples, with ReLU (Rectified Linear Unit) activation function, followed by (ii) dropout layer with 0.2 dropout rate;
2. Flattening layer;
3. Output layer with one neuron.

At input of discriminator is a generated/real signal with length of 1,500 samples and estimation if this signal is generated or real (0/1) is at output.

The generator (Fig. 7), on the other hand, has the following architecture:

1. Input latent vector of 1,000 random numbers;
2. One dense layer with 100 neurons and sigmoid activation function;
3. Fully connected output layer with 1,500 neurons.

The generator generates 1,500 samples based on 1,000 random numbers.

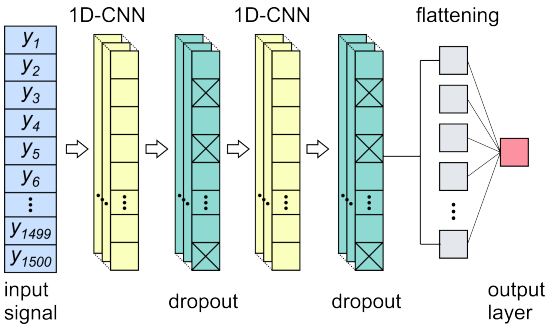


Fig. 6. The architecture of discriminator

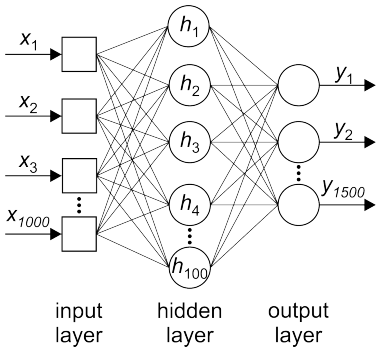


Fig. 7. The architecture of generator

For GAN training a batch of 2,000 real signals $\boldsymbol{s_i}$, $i \in [0, 1999]$, with length of 1,500 samples each, are extracted from training dataset in the following way:

$$\boldsymbol{s_i} = [x_{50i+1}, x_{50i+2}, \ldots x_{50i+1500}] \quad (1)$$

exploiting a total of 101,450 samples acquired from DisEPP. During training 500 epochs were employed.

Figure 8 presents an example of signal obtained using the trained generator. This signal is similar to the excerpt of signal obtained from real process (Fig. 5).

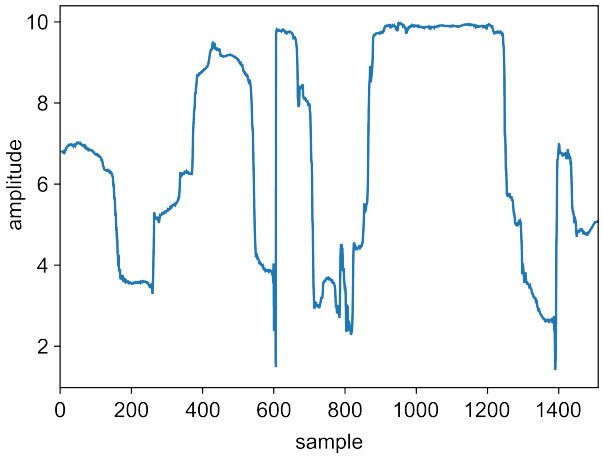


Fig. 8. An example of signal obtained using the generator

 ISBN 978-86-7466-930-3

B. *IDS creation using generated and acquired signals*

Following the procedure presented in Section III, two IDS were created: one using generated signals, and the other using real signals obtained from DisEPP. The same preprocessing procedure is applied to both signals, where the buffers of v=16 samples were used. The employed FIR filter in signal processing has pass band of [0, 0.11π], stop band of [0.35π, π], transition region in between. The filter is composed of 11 coefficients ([0.020243, 0.023017, 0.054189, 0.10397, 0.12557, 0.12344, 0.12557, 0.10397, 0.054189, 0.023017, 0.020243]) that are generated by the Parks-McClellan algorithm. After applying the filter, the signal is normalized by its maximum value. During models training, the whole datasets (signals structured into ordered pairs) are divided into training, validation and test part, with a share of 70/10/20%, respectively. Model training was performed through 5 epochs with Adam optimizer (learning rate of 0.001) and the mean squared error (MSE) cost function.

Following the procedure presented in Section III, two CNN models were developed:

1. based on signals generated using GAN where a total of 266 signals with 1,500 samples containing a total of 391,818 ordered pairs;
2. based on real signal from DisEPP that contains 399,000 samples corresponding to 398,973 ordered pairs.

For both signals (real and generated) the models with the same architecture were obtained. This architecture is composed of the following layers:

- 1D-CNN (4 filters, kernel size=2)
- 1D-CNN (8 filters, kernel size=2)
- Max pooling (pooling rate=2)
- 1D-CNN (16 filters, kernel size=2)
- 1D-CNN (16 filters, kernel size=2)
- Max pooling (pooling rate=2)
- Flattening
- Dense (30 neurons)
- Dense (1 neuron).

and it has a total of 2865 trainable parameters. The model was trained in Python v3.8.5 using a Spyder with TensorFlow v2.3.0 in the background.

In the online part of the algorithm, the detection threshold is calculated as a sum of the mean (μ) value and the triple standard deviation (σ) of discrepancies between received and estimated values over the testing data:

$$T=\mu+3\sigma \tag{2}$$

and it was T=0.00941 for the IDS based on generated and T=0.00956 for IDS based on real-world data.

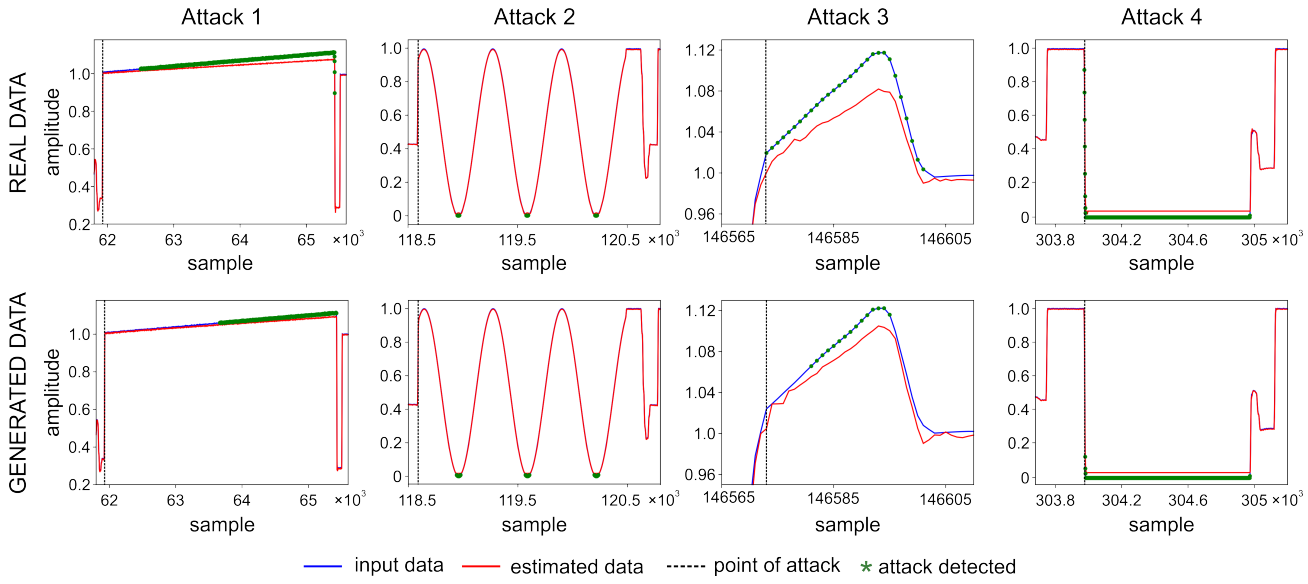


Fig. 9. Detected cyber-attacks

If the discrepancy between real and estimated value is higher than the threshold for z=15 consecutive signal samples, then the attack is detected.

To test the performance of the developed IDSs they were simulated using in Python using a number of the designed attacks. In this paper, we present and compare the performances of IDSs using four attacks with different shapes and duration. Attacks 1 and 3 increase signal value linearly by 0.00003 and 0.005 per sample, respectively, where in attack 1 random noise was introduced as well. Attack 2 utilizes sine function to generate signal value, whereas in attack 4 the signal value is set to 0 for predefined time period.

The results of attacks detection using IDSs obtained based on generated and real data are presented in Fig. 9. In this Figure input data and their estimation are shown in blue and red lines, respectively, the start of the attack is represented with a black dashed line, whereas the moments when the attack was detected are marked with green markers. Both

 ISBN 978-86-7466-930-3

models proved to be equally effective and successfully detected all attacks without false positives.

It can be noticed from the Fig. 9 that the model based on the real data detected attack 1 earlier than the model that used GAN-generated data for training. On the other hand, for attacks 2, 3, and 4, the difference between the detection moments is negligible.

V. Conclusion

In this paper, we have explored the possibilities for utilization of GAN based data augmentation in the design of IDS for ICS using an example of electro-pneumatic positioning system DisEPP based on smart devices. Using a limited number of real signal samples obtained from the DisEPP, an amount of data sufficient for deep learning based generation of IDS was generated. Using the previously developed attack detection method based on CNN, two IDS were created: one using generated signals, and the other using real signals obtained from DisEPP. To evaluate the performance of the created IDSs, a number of attacks have been created, of which four are presented in the paper. As presented in the paper, the IDS based on generated data was able to successfully detect all cyber-attacks without false positives. It presented the similar performances as IDS based on original data not only in terms of a number of detected attacks, but also in terms of attack detection latency, thus confirming that GAN augmented data can be successfully utilized for the generation of semi-supervised data based IDS in ICS.

In the future, we plan to extend our work to additional datasets that contain a higher number of signals and attacks. Furthermore, our research efforts will be directed to the application of different types of GAN and a comparison of their performances.

Acknowledgment

This research was supported by the Science Fund of the Republic of Serbia, grant No. 6523109, AI-MISSION 4.0 as well as by the Ministry of Education, Science and Technological Development of the Serbian Government under the contract No. 451-03-68/2022-14/200105.

References

[1] H. Kagermann, W. Wahlster, J. Helbig, *Recommendations for implementing the strategic initiative INDUSTRIE 4.0*, 2013. [Online]. Available: http://www.acatech.de

[2] M. R. Asghar, Q. Hu, S. Zeadally, "Cybersecurity in industrial control systems: Issues, technologies, and challenges," *Computer Networks*, vol. 165, article no. 106946, 2019.

[3] D. Upadhyay, S. Sampalli, "SCADA (Supervisory Control and Data Acquisition) systems: Vulnerability assessment and security recommendations," *Computers & Security*, vol. 89, article no. 101666, 2020.

[4] K. Stouffer, J. Falco, K. Scarfone, *Guide to industrial control systems (ICS) security*. NIST special publication, 2015.

[5] Y. Xu, Y. Yang, T. Li, J. Ju, Q. Wang, "Review on cyber vulnerabilities of communication protocols in industrial control systems," IEEE Conference on Energy Internet and Energy System Integration (EI2), pp. 1-6, IEEE, Beijing, China, Nov. 2017.

[6] Industrial Control Systems Cyber Emergency Response Team, Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies, 2016, Available: https://www.cisa.gov/uscert/sites/default/files/recommended_practices/NCCIC_ICS-CERT_Defense_in_Depth_2016_S508C.pdf, Accessed on: Mar. 2022.

[7] L. K. Carvalho, Y. C. Wu, R. Kwong, S. Lafortune, "Detection and mitigation of classes of attacks in supervisory control systems," *Automatica*, vol. 97, pp. 121-133, 2018.

[8] Z. Jakovljevic, V. Lesi, M. Pajic, "Attacks on Distributed Sequential Control in Manufacturing Automation," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 775-786, 2021.

[9] M. Elnour, N. Meskin, K. Khan, R. Jain, "A Dual-Isolation-Forests-Based Attack Detection Framework for Industrial Control Systems," *IEEE Access*, vol. 8, pp. 36639-36651, 2020.

[10] S. Sapkota, A. K. Mehdy, S. Reese, H. Mehrpouyan, "Falcon: Framework for anomaly detection in industrial control systems," *Electronics*, vol. 9, no. 8, article no. 1192, 2020.

[11] A. Al-Abassi, H. Karimipour, A. Dehghantanha, R. M. Parizi, "An ensemble deep learning-based cyber-attack detection in industrial control system," *IEEE Access*, vol. 8, pp. 83965-83973, 2020.

[12] G. Raman MR, N. Somu, A. Mathur, "A multilayer perceptron model for anomaly detection in water treatment plants," *International Journal of Critical Infrastructure Protection*, vol. 31, article no. 100393, 2020.

[13] M. Kravchik, A. Shabtai, "Detecting Cyber Attacks in Industrial Control Systems Using Convolutional Neural Networks," Proceedings of CPS-SPC 18 Conference, pp. 72-83, Toronto, Canada, Oct. 2018.

[14] D. Nedeljkovic, Z. Jakovljevic, "CNN based method for the development of cyber-attacks detection algorithms in industrial control systems," *Computers & Security*, vol. 114, article no. 102585, 2022.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[16] J. Brownlee, *Generative Adversarial Networks with Python: Deep Learning Generative Models for Image Synthesis and Image Translation*. Machine Learning Mastery, 2019.

[17] F. Chollet, *Deep learning with python*. Manning, Shelter Island, NY, USA, Nov. 2017.

[18] NXP Semiconductors N.V. (2009, Feb.), "LPC1769/68/66/65/64/63 32-bit ARM Cortex-M3 microcontroller," [Online]. Available: https://www.nxp.com/docs/en/data-sheet/LPC1769_68_67_66_65_64_63.pdf, Accessed on: Mar. 2022.

[19] Microchip Technology Inc. (2008) "MRF24J40MA 2.4 GHz IEEE Std. 802.15.4TM RF Transceiver Module," [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/70329b.pdf, Accessed on: Mar. 2022.

 ISBN 978-86-7466-930-3

Comparison of SLAM algorithms on omnidirectional four wheel mobile robot

Slaven Petković, Lazar Milić, Milutin Nikolić, Dragiša Mišković, and Mirko Raković

***Abstract*—In this article we present a comparative analysis of various SLAM algorithms. We compared robot trajectories computed by three ROS-based SLAM algorithms to a reference trajectory obtained from Vicon motion capture system. For data acquisition purposes we used mobile robot with four omnidirectional (*mecanum*) wheels. Our mobile platform was equiped with following sensors: 3D lidar, a RGB-D camera and motor encoders. Experiments were conducted indoor in an office environment. Acquired dataset was used as an input data for all algorithms that we tested. Following algorithms have been taken into account: Livox Mapping, RTAB-Map and Cartographer.**

***Index Terms*—Simultaneous Localization and Mapping, SLAM, ROS, Mobile robot**

I. Introduction

ONE of the most important tasks for autonomous mobile robots is to create a consistent map of unknown environment and to determine its location inside that map. This problem is known as Simultaneous Localization and Mapping (SLAM) and it is considered a difficult problem, because robot needs good estimate of its location in order to create a valid map, but at the same time robot needs consistent map to determine its location.

Problem was first defined back in 1986 [1]. There are two main approaches for solving SLAM problem: probabilistic approach and non-probabilistic approach. The probability methods are based on Bayesian estimation method. Many methods were developed, such as SLAM based on Kalman Filter, Extended Kalman Filter [2], Particle Filter [3], Rao-Blackwellized Particle filter [4], etc.

Nowadays there are many different methods for solving SLAM problem. In this article we focus on SLAM algorithms that are available as an *open-source* ROS (Robot Operating System) [5] package and that have support for 3D lidar and/or RGB-D camera. For purpose of acquiring data, mobile robot with four omnidirectional wheels was used and all experiments were done indoor.

II. Related Work

In this section, we provide an insight into papers that deal with the comparison of SLAM algorithms. Paper [6] provides benchmark for two popular SLAM algorithms: RTAB-Map and RGBD SLAM. Paper [7] shows comparison results of three different mapping approaches. Comparison between three modern VSLAM approaches : RTAB-Map, ORB-SLAM3 and OpenVSLAM is presented in [8]. In article [9] authors show comparison of trajectories computed by various ROS-based SLAM systems in office environment. This article presents a different approach to comparing SLAM algorithms.

S. Petković, L. Milić, M. Nikolić, M. Raković are with the Faculty of Technical Science, University of Novi Sad, (e-mail: rakovicm@uns.ac.rs). D. Mišković is with the Institute for Artificial Intelligence Research and Development of Serbia, Novi Sad, Serbia

III. Compared SLAM algorithms

A. RTAB-Map

RTAB-Map (Real-Time Appearance-Based Mapping) [10] is an open-source library released in 2013 and is still in the development. It is a graph-based SLAM approach with memory management available as *rtabmap_ros* ROS package. RTAB-Map is very flexible SLAM approach because it supports wide range of input sensors such as odometry from any source, RGB-D or stereo cameras and optionally 2D/3D lidar data. Package consists of main *rtabmap_ros/rtabmap* node, two nodes for visual odometry (*stereo_odometry* and *rgbd_odometry*) and lidar odometry node (*icp_odometry*). As RTAB-Map supports inputs from multiple different sensors, data obtained from these sensors needs to be synchronized in order to create a valid map. There are two types of synchronization in RTAB-Map: exact synchronization and approximate synchronization. RTAB-Map can be configured with large number of parameters whose description can be found in [11].

B. Cartographer

Google's Cartographer is lidar graph-based SLAM approach. Cartographer is used for indoor mapping and it supports 2D and 3D lidar data as well as odometry data. In the process of creating map Cartographer uses lidar scans or point cloud to create sub-maps and when loop-closure is detected it runs pose optimization in order to minimize error. First part of this process is managed by Cartographer's local SLAM subsystem (also called frontend), while optimization is done using Cartographer's global SLAM subsystem (also called backend). Detailed description of Cartographer can be found in [12]. On official web page [13] one can find algorithm explanation, description of all parameters and tips for parameter adjustment.

C. Livox Mapping

Livox mapping is a SLAM algorithm developed for creating a map using only Livox lidars. It is available as *livox_mapping* ROS package which supports multiple Livox 3D lidars. Algorithm also uses odometry data, for example wheel odometry. In the development of this package, authors reference to [14]-[16].

 ISBN 978-86-7466-930-3

IV. System setup and Data set acquisition

In this section, we will present the hardware and software used to conduct the experiment. Experiment consists of three main modules: (1) Vicon system for motion capture, (2) a mobile robot for acquiring sensor data and (3) a notebook for running and comparing SLAM algorithms.

A. Vicon system

Vicon system is a highly accurate motion capture system. This system can be used in multiple applications, like robot tracking, human motion capture, etc. In our experiment, the main task of this system was to capture robot's motion and compute trajectory of the robot. This trajectory was used as a reference trajectory to compare and evaluate different SLAM algorithms. Our indoor testing environment with Vicon system and mobile robot is shown in Figure 1. In this experiment Vicon system consisted of (1) seven Vicon MX T-20S cameras, (2) Vicon MX Giganet and (3) desktop computer with Vicon Nexus software used for data processing. Calibration of cameras is done using Vicon Active Wand calibration device and Vicon Nexus software. In this experiment, the operating frequency of Vicon cameras was set to 200 Hz.

Fig. 1: Experimental setup with Vicon cameras and mobile robot

In order to record the movement of the mobile robot, it was necessary to place reflective markers on the mobile robot, which can be tracked by motion capture system. Position and orientation of these markers defines coordinate system of tracking object created in Vicon Nexus software. The orientation of the coordinate system depends on the order in which the markers are selected in Vicon Nexus software. Figure 2 shows three markers placed on the mobile robot. To make it easier to compare data later, the position and orientation of the markers on the mobile robot is chosen to match *base_link* coordinate system defined in robots URDF model (*Unified Robot Description Format*).

B. Mobile robot

Mobile robot used in this experiment was a four wheel omnidirectional mobile robot (Figure 3). This construction with

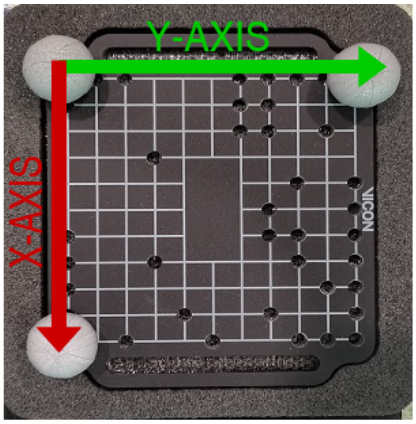


Fig. 2: Reflective markers used to track robot motion by Vicon system. Position of markers determines position and orientation of coordinate system of tracked object

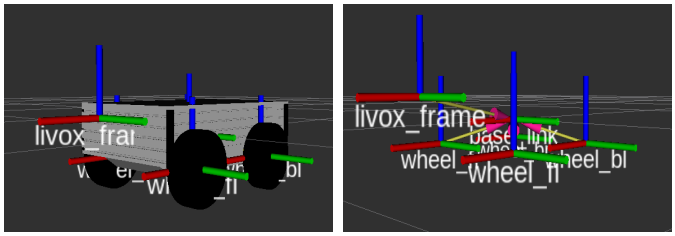
omnidirectional wheels (*mecanum wheels*) enables it to move in multiple directions and change direction rapidly. Mecanum wheel consists of k rollers made of rubber positioned at 45 degrees angle offset from the wheel rotation around its circumference. Paper [17] describes geometry and kinematics of mecanum wheels. Main problem with this type of mobile robots is that robot's wheels slip, and odometry based on encoders from wheels can not be considered as reliable.

Fig. 3: Omnidirectional four-wheeled mobile robot

Running SLAM algorithms on ROS requires a file that describes robot's physical state to ROS. Creating robot model was done using URDF. URDF is an XML file format used to describe all components of a robot. Figure 4 presents model of our mobile robot and positions of coordinate systems presented in RViz.

Robot's software runs on Ubuntu 18.04 operating system and ROS Melodic. Following sensors were mounted on mobile robot in this experiment: 3D lidar, RGB-D camera and wheel

 ISBN 978-86-7466-930-3



(a) Robot visualization in RViz (b) Robot coordinate systems

Fig. 4: Mobile robot model in RViz

encoders. Detailed information about robot configuration is presented in Table I.

Hardware	
CPU	AMD Ryzen 7 2700u
GPU	AMD Radeon Vega 10 Graphics
RAM	16 GB DDR4
Sensors	
Lidar	Livox MID-70
Camera	Intel RealSense D435i
Software	
Operating System	Ubuntu 18.04 Bionic Beaver
ROS	ROS Melodic Morenia

TABLE I: Hardware and software specifications of mobile robot

C. Data processing

In order to get the most valid comparison results, the goal was to record data from robot sensors to one file, so every SLAM algorithm tested in this experiment could use the same input data. For this reason we run offline all the SLAM algorithms that are evaluated. Detailed information about computer configuration that was executing SLAM algorithms is shown in II.

Hardware	
CPU	Intel Core i7-10750H
GPU	NVIDIA RTX 2060 6GB
RAM	16 GB DDR4 3200 MHz
Software	
Operating System	Ubuntu 18.04 Bionic Beaver
ROS	ROS Melodic Morenia

TABLE II: Hardware and software specifications of computer used for data processing

V. Metrics and data comparison

In this section, we give an insight into how the Vicon system and ROS represents robot pose. We also present metrics for analyzing performance of SLAM algorithms by comparing trajectories and robot poses generated by each SLAM algorithm to a ground truth trajectory and robot poses computed by Vicon system. We did not compare generated maps. The idea was to compare predicted robot pose to an actual robot pose in every moment in time. As robot pose in plane consists of three components, x and y coordinate and rotation around z-axis, we compared each component of robot motion separately. Then the error for all three components of robot motion was calculated.

A. Data representation

1) Vicon system: Vicon Nexus software is able to export data in number of different formats. We have chosen to work with *.csv* file. Table III shows an example of *.csv* file generated by Vicon Nexus software. Values *RX*, *RY* and *RZ* represent rotation of robot's base coordinate frame around x, y and z axis respectively, while *TX*, *TY* and *TZ* represent position of robot's coordinate frame in x, y and z direction respectively with respect to the reference frame. All values are expressed in relation to the origin of the coordinate system defined by the position of the Vicon Active Wand during calibration. Frame column was used to calculate timestamp.

Frame	Sub Frame	RX	RY	RZ	TX	TY	TZ
number	number	[deg]	[deg]	[deg]	[mm]	[mm]	[mm]

TABLE III

2) SLAM algorithms: The ability of Robot Operating System to record data from any available topic was used for generating data set. Recorded data was saved to *.bag* file. ROS represents robot pose as *geometry_msgs/Pose* ROS message and it also enables us to export in *.csv* file. Unlike Vicon system, ROS uses Quaternion to display robot rotation.

B. Data comparison

Comparing two trajectories requires them to consist of same number of points (robot poses) and to be synchronized in time. In our case, neither of these two conditions was met.

As previously said, we collected data from sensors mounted on mobile robot and run algorithms offline using collected data to generate test data set. Reference trajectory was computed by Vicon system. These systems are not synchronized in time.

First problem was how to equalize the beginnings of two compared trajectories in time. In order to do that, we decided to find moment in time in which robot made a movement for 1 mm in x or y direction.

Second problem was that every ROS algorithm publishes messages about robot pose at different rate and Vicon system frequency was 10 to 20 times greater than frequency at which ROS algorithms publish messages. In order to deal with this problem, there were two possibilities: to reduce the size of reference data set or to expand test data set. We opted for the latter and decided to do an interpolation over data generated by SLAM algorithms.

C. Metrics

Interpolation was done by time for every component of robot motion: $x(t)$, $y(t)$ and $\theta(t)$. Figure 5 shows example of interpolation in case of function $x(t)$.

As previously sad, SLAM algorithms publish messages about robot pose less frequently than Vicon system. In this paper interpolation was used to calculate new value in test data set for every moment in time of reference data set. Let's observe point (t_i, x_i) from the reference data set. In order to calculate new value we decided to find two points in the test data set which were closest in time to the observed point. On figure 5 points A(t_j, x_j) and B(t_{j+1}, x_{j+1}) represent two closest points. Equation of line connecting points A and B was calculated. Inserting value of t_i into new equation, interpolated value x_{interp} is calculated.

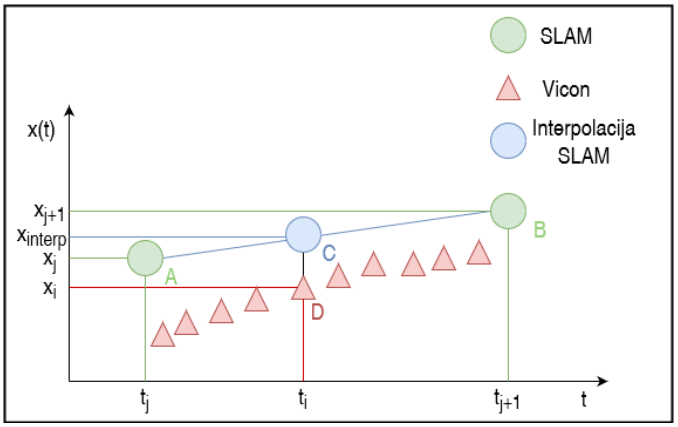


Fig. 5: The figure illustrates an example of interpolation used to generate new data in test data set

Data processing and comparison was done using the Python 3 programming language and the *NumPy* library, as one of the best open-source tools for working with data and numerical calculations.

VI. Results and Analysis

In this section we present results of our experiment. Results are presented graphically and numerically. For visualizing results, we used Python programming language and *Matplotlib* library. Results are presented in four graphs for each tested SLAM algorithm. First figure contains estimated trajectory (shown in red) computed by SLAM algorithm and reference trajectory computed by Vicon system (shown in green). Other three figures show comparison of *x(t)*, *y(t)* and $\theta(t)$ functions respectively. Table IV presents numerical results of our experiments. Error is represented as RMSE (*Root Mean Square Error*), standard deviation, mean absolute error and maximum absolute error.

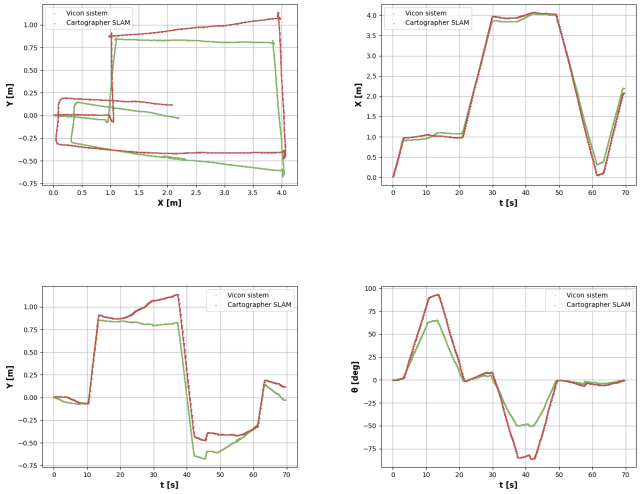


Fig. 6: Trajectory recovered from Cartographer SLAM (red) vs Vicon system reference trajectory (green)

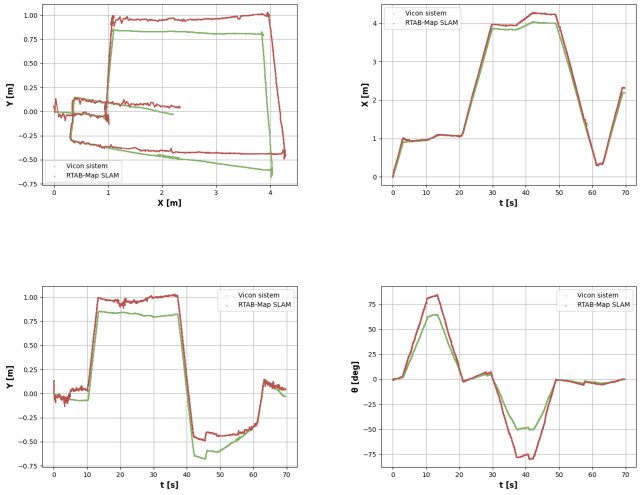


Fig. 7: Trajectory recovered from RTAB-Map (red) vs Vicon system reference trajectory (green)

VII. Conclusion

In this paper, we have presented a method for analyzing performance of ROS-based SLAM algorithms that compares trajectories computed by SLAM algorithm to a reference trajectory obtained from motion capture system. We proposed a metric for calculating error in estimated trajectory. This approach allows us to compare algorithms that use different sensor information.

Since omnidirectional wheels are slipping during the motion of robot, the localisation of the robot has to be based on sensory system that is not measuring motion of wheels directly. Therefore, the robot is build with 3D Lidar and RGB-D camera as sensors for localisation of the robot and mapping of the environment. To better determine what algorithm is the best for sensory system that our robot has, we evaluated three

 ISBN 978-86-7466-930-3

Benchmarking results

SLAM Algorithm	RMSE [cm]	Standard deviation [cm]	Mean absolute error [cm]	Maximum Absolute Error [cm]
Comparison of error in x direction				
RTAB-Map	1.2518514	7.6750762	9.8897118	27.0362502
RTAB-MAP Visual odometry	1.5254185	7.9799980	13.0003765	29.3477365
Google Cartographer	1.1652417	7.1982585	9.1631812	33.8634761
Livox Mapping	1.6213981	9.1069006	13.4148259	38.366215
Comparison of error in y direction				
RTAB-Map	1.3296018	6.5810257	11.5531029	23.8609514
RTAB-MAP Visual odometry	2.3063390	8.1394842	13.0003765	35.0320307
Google Cartographer	1.6002339	10.2693004	21.5793597	37.0653832
Livox Mapping	1.1607506	6.6658583	9.5026594	24.9565594
Comparison of error in rotation				
	RMSE [°]	Standard deviation [°]	Mean absolute error [°]	Maximum Absolute Error [°]
RTAB-Map	12.1708190	9.2212436	7.9433937	29.8499295
RTAB-MAP Visual odometry	12.4845912	9.2257357	8.4113505	29.9491947
Google Cartographer	16.1672031	12.0188858	10.8131790	37.3971806
Livox Mapping	19.1045268	13.6542612	13.3620393	46.4193679

TABLE IV: Benchmarking results

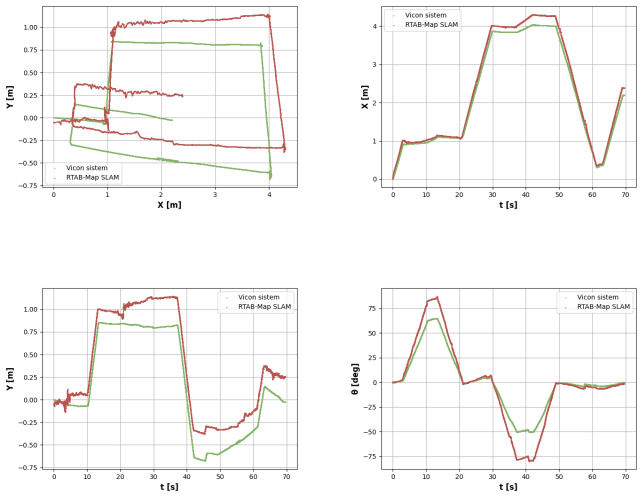


Fig. 8: Trajectory recovered from RTAB-Map visual odometry (red) vs Vicon system reference trajectory (green)

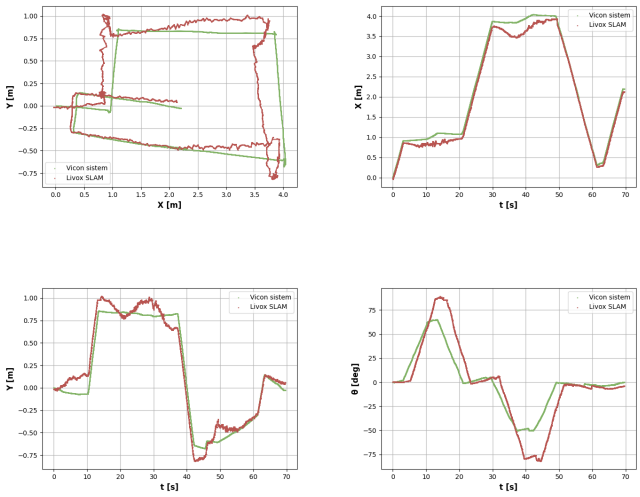


Fig. 9: Trajectory recovered from Livox Mapping (red) vs Vicon system reference trajectory (green)

up-to-date algorithms. Based on benchmark results we can see different algorithms provide different best performance. In most cases, RTAB Map has shown the best results compared to other algorithms.

REFERENCES

[1] Durrant-Whyte, Hugh, and Tim Bailey. "Simultaneous localization and mapping: part I." IEEE robotics & automation magazine 13.2 (2006): 99-110.

[2] Bailey, Tim, et al. "Consistency of the EKF-SLAM algorithm." 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2006.

[3] Montemerlo, Michael, et al. "FastSLAM: A factored solution to the simultaneous localization and mapping problem." Aaai/iaai 593598 (2002).

[4] Grisetti, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters." IEEE transactions on Robotics 23.1 (2007): 34-46.

[5] Quigley, Morgan, et al. "ROS: an open-source Robot Operating System." ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.

[6] Kasar, Amey. "Benchmarking and comparing popular visual SLAM algorithms." arXiv preprint arXiv:1811.09895 (2018).

[7] Burgard, Wolfram, et al. "A comparison of SLAM algorithms based

on a graph of relations." 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2009.
[8] Merzlyakov A, Macenski S. A Comparison of Modern General-Purpose Visual SLAM Approaches. arXiv preprint arXiv:2107.07589. 2021 Jul 15.
[9] Filipenko, Maksim, and Ilya Afanasyev. "Comparison of various slam systems for mobile robot in an indoor environment." 2018 International Conference on Intelligent Systems (IS). IEEE, 2018.
[10] Labbe, Mathieu, and François Michaud. "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation." Journal of Field Robotics 36.2 (2019): 416-446.
[11] https://github.com/introlab/rtabmap/blob/master/corelib/include/rtabmap/core/Parameters.h
[12] Hess, Wolfgang, et al. "Real-time loop closure in 2D LIDAR SLAM." 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016.
[13] https://google-cartographer-ros.readthedocs.io/en/latest/index.html
[14] Zhang, Ji, and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time." Robotics: Science and Systems. Vol. 2. No. 9. 2014.
[15] Zhang, Ji, and Sanjiv Singh. "Enabling aggressive motion estimation at low-drift and accurate mapping in real-time." 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017.
[16] Zhang, Ji, Michael Kaess, and Sanjiv Singh. "On degeneracy of optimization-based state estimation problems." 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016.
[17] Gfrerrer, Anton. "Geometry and kinematics of the Mecanum wheel." Computer Aided Geometric Design 25.9 (2008): 784-791.

 ISBN 978-86-7466-930-3

Natural Non-Invasive Human-Machine Interface Based on Hand Gesture Recognition

Jelena Rodić, Darko Golubović, Nikola Knežević, Kosta Jovanović

***Abstract*— In recent years human-machine interfaces have been identified as an important aspect for enabling safe and efficient human-robot collaboration. In the same period of time, deep learning has made great progress in image classification problems with the evolution of convolutional neural networks This paper presents a hand gesture classification module as a non-invasive natural human-machine interface that exploits deep learning technology. There were various approaches for this task in the past, such as lookup tables, detection of key-point positions of fingers, classic neural networks, etc. This paper implements VGG16 convolutional neural network to solve the task of hand gesture. To capture an image, we use leap motion sensor which is cheap and can work in challenging light conditions, because it uses infra-red emitters to lighten the object. Thus, this approach is useful for factories and production lines. Another contribution of this paper is an extensive database consisting of 20 000 images.**

***Index Terms*— Human-machine interface; Hand gesture; Convolutional Neural Networks; Leap motion sensor; VGG16**

I. Introduction

Enhanced by developments in technology, numerous jobs are being transferred into the domain of robots and sophisticated machines [1, 2, 3]. These production lines perform various tasks. However, because market requirements are changing daily, there is a need for quick adaptation of these lines. Therefore, some manufacturers have developed flexible robotic cells that can serve multiple types of processes [4]. On the other hand, humans can adapt to changes in requirements quickly. This quality makes them valuable considering the dynamic of market needs. Combining human flexibility and robot reliability and consistency could give us a solution to the challenge the market sets without sacrificing the output volume.

The application of such a system is displayed in assembling fiscal cash registers. The process itself requires human involvement because the parts needed for assembly are small and the assembly process is delicate (mounting flat cables, soldering capacitors for PCBs, inserting tape for thermal printers). This process could be significantly less demanding for the operator by using novel technologies.

Firstly, it is possible to show the operator instructions for assembling a specific device using adaptive screens. Secondly, collaborative robots can be used to deliver the right part for the current production phase on time, which would increase the overall speed of production.

Thirdly, it is possible to gain an insight into worker performance (focus, monotony, fatigue) during tasks using EEG headphones. With this approach, the worker could switch assignments occasionally, and stay interested and productive.

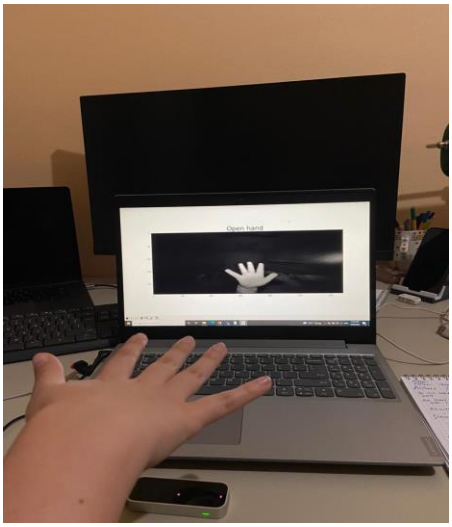


Fig. 1. System setup. Leap Motion sensor connected to PC for classification of hand gestures.

Also, if a person is engaged in dangerous work, this approach can prevent injuries from occurring when their focus is low and fatigue high. In addition to assistive devices, it is necessary to enable intuitive communication between humans and the rest of the system. For this purpose, a Leap Motion sensor is used (Fig. 1).

This sensor can detect hand gestures and provide human-machine interaction. There were a couple of solutions providing the human-machine interface but, some of them were invasive (wearing sensor gloves) or non-intuitive (complex system of buttons and taskbars). Using this type of sensor non-invasive method of human-machine interface is introduced. In this way communication between humans and machines is alleviated and it is more receptive to humans.

In this paper, we present a system for classifying human hand movements. For this purpose, it was necessary to collect the appropriate database, select the neural network architecture that will perform the classification, and finally test and implement the application that works in real-time.

II. Previous work

Paper [5] from the University of Padua implements hand movement recognition. The classification was done over ten basic hand gestures. Also, the paper focuses on static gesticulation using leap motion controllers and kinetic devices. Microsoft's Kinect camera provides 3D images, while leap motion provides only key points in 3D. Here the authors explain that the images from ordinary cameras are 2D variants of the 3D position of the hand, so there is a loss of information.

 ISBN 978-86-7466-930-3

By providing 3D finger positions leap sensor can give more meaningful measurements. With the introduction of Time-of-Flight cameras for widespread use, 3D representations of objects have become more accessible to use. However, recognizing any movements that are not the most basic is still too complex for this type of sensor (leap motion), the paper states. The authors decided to unite these two types of controllers and try to get a more accurate classification. The problem is to calibrate these two devices. They achieved this by combining the fingertips from leap motion and the peaks on the depth camera that represent the fingers. Attributes used are the number and position of the fingers, the center of the palm and the orientation of the hand, angles at which the fingers stand, and the distances between the fingers. The problem occurs when the fingertips touch because sometimes they are detected as one finger. Furthermore, rings, bracelets, etc. can spoil the accuracy too. After PCA analysis, features were input into a multiclass SVM classifier with a Gaussian kernel. The best accuracy authors achieved is 80.9%, using only the leap sensor. After they inserted the data from Kinect, the accuracy jumped to 96%.

Paper [6] deals with the recognition of dynamic hand movements. The attributes used in this paper can be divided into two categories: static and dynamic. Static ones are based on the positions of the fingers and palms, ie. at the relative distances between them. Distances between the tips of adjacent fingers, and between the center of the palm and the tip of the finger are defined as attributes. For example, the Ok symbol is represented by the distance between the thumb and forefinger as a good attribute. Dynamic attributes are defined by finger and palm speeds to detect moving patterns. Those patterns are complicated, but the velocities of the essential points are given by the sensor itself, which makes the job easier. Examples of dynamic attributes used here are the translation of the whole hand (when the palm and fingers move at the same speed along some axis, without rotation), rotation of the hand (when the palm rotates), the precession of the hand (when the palm moves in a circle), swipe index finger, etc. The results of this work show that when the controller detects fingers accurately, the classification methods themselves work accurately. The authors noticed that tracking the fingertips of the middle and little fingers is not very stable. This can cause some problems in inference.

Paper [7] deals with the application of leap controllers to realize a virtual museum for free. Interaction with the virtual world using hand gestures has been suggested. It is necessary to classify the positions of the hand to prevent unwanted moves. First, data based on x, y, and z coordinates of fingertips was obtained, and then attribute extraction was performed. The authors used information on how far the fingertips are above the plane of the palm. They also used information on angles between the fingers and the palm. The classification method used was K nearest neighbors (KNN). They state that the accuracy obtained in other papers is about 80% for the same subjects and about 70% for new ones. The exact accuracy they obtained was 99%, but it was not said how they chose the test set.

Paper [8] was done at the University of Malaysia and deals with the control of drones based on hand movements read from leap sensors. The idea is that leap motion reads the gesture, sends it to the Arduino for recognition, and then sends the command to the drone to control. Three throttle commands are classified - lift, pitch, and roll. The idea was to simplify communication with the drone, which is currently complicated since there are numerous handles, buttons, etc. Therefore, the new commands were up and down, tilt back and forth, and tilt left and right. It is said that the drone worked well when the commands were given clearly. No accuracy or classification methods were stated.

In [9] authors implemented automation of design commands in CAD, Solidworks, and Catia software. It is stated that there is a lack of tools for some very intuitive modifications. The authors wanted to replace the commands given by the mouse and keyboard. For example, the review mode requires translation, rotation, and scaling. The scaling command is the distance between two index fingers. The rotation command is rotating the fist. In the end, a grab, and release command for translation are performed based on the distance between the index fingers. Paper does not use machine learning for classification but only distance mapping. If a command cannot be mapped to the table, then it is omitted. The accuracy they get is 80%.

Paper [10] dealt with a slightly different task. They wanted to recognize handwritten symbols in the air based on the movement of the hand. The key points were sampled at a rate of 100 frames per second. The attributes taken are the positions and velocities of the key points, the angles at which the fingers stand, etc. Finally, the classification was performed using convolutional neural networks. The accuracy they obtained is 92.4%.

Paper [11] deals with the dynamic movements of human hands. Motion information is converted to color images. The conversion of movement into an image is done by tracking the finger position in time. Each finger is assigned to a distinct color, and the intensity of the color changes depending on the time. The authors used the ResNet-50 for the classification. The accuracy they get is about 80%.

III. METHOD

First, we need to define the problem we want to solve. Namely, for the needs of working on the production line in factories, it is convenient to use only commands that do not include pressing the screen and buttons when worker's hands are dirty, or the screen can't detect users' input caused by wearing the safety gloves or other equipment. For our case, it was necessary to select 4 hand commands that could be classified with high precision using the leap motion sensor. Some crucial steps in solving this problem were:

- Obtain or form an adequate dataset that has different commands given by hand movements. We decided to work on images because they are more reliable than the detection of key points from Leap sensors. Also, convolutional neural networks are powerful in image

 ISBN 978-86-7466-930-3

classification, so this method should give promising results.

- Choose 4 distinct commands, which will be able to be easily distinguished from the images returned by the Leap sensor.
- Choose the adequate architecture of the convolutional neural network and tune the hyperparameters. Assess the probability of misclassification on the test set.
- Eventually do a live simulation with a real sensor where after the first detection of the hand in the frame three frames would be captured at 0.33s interval. Then each of these three frames would be passed through a neural network for classification, and finally, a final decision would be made based on a majority vote.

A. Leap Motion Sensor

There are several sensors on the market that handle motion detection. The Leap Motion sensor [12] stands out for its price (it is cheap) and accuracy. Leap Motion Controller is an interactive device specializing in detecting hand movements and finger location based on infrared light emitters and two cameras that receive reflected IR waves. The Leap Motion architecture can be seen in Fig. 2.

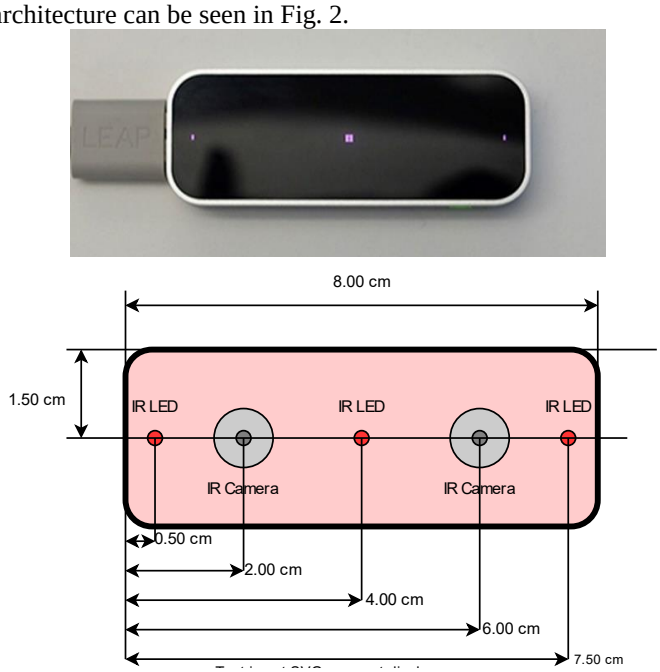


Fig. 2. Leap motion sensor, from [14]

Its field of vision is up to 1m away. There is also a skeletal model of the human hand, which represents five fingers divided into phalanges. We can also obtain a raw image from the sensor, which is the one we used.

B. Database

First approach was to try to train the network on different publicly available datasets. The best result we got was with the database [15]. The problem with this database is that it did not contain various heights or angles on which the hand can be, thus making this network prone to errors once the hand is not perfectly positioned. Considering the application of this paper, we wanted to develop a system that is able to work in real life conditions, thus making this database inadequate for our task. On our versatile test dataset, this network had the accuracy of 91.5%.

After realizing that publicly available databases are not extensive enough to meet our needs, we decided to develop our own dataset using the Leap Motion sensor. Our training set consists of four classes of hand gestures. Each class consists of 5000 images. There is an equal part of men and women in the hand dataset. Also, there are 50% left-hand images in the database. Furthermore, images were obtained at various lighting conditions, angles, and heights. The last two turned out to be important for the network to generalize well. We trained our network on one publicly available data set which had only photos of hands perpendicular to the sensor. Also, all the pictures were formed with a constant distance between the hand and the sensor. This plays a key role because the camera on the leap sensor is in a wide range. That slight change in the distance of the camera results in a tremendous change in the size of the object in the final image. As a result, the same architecture performed significantly worse than when it was trained on our custom dataset. This shows us the importance of a well-rounded dataset, which achieved accuracy of 99.3%. We will compare these results in detail in the section Results.

In the end, we used an unseen person to obtain the test set. Also, the test set contains images taken from different angles and heights. Here are examples from the dataset (Fig. 3):

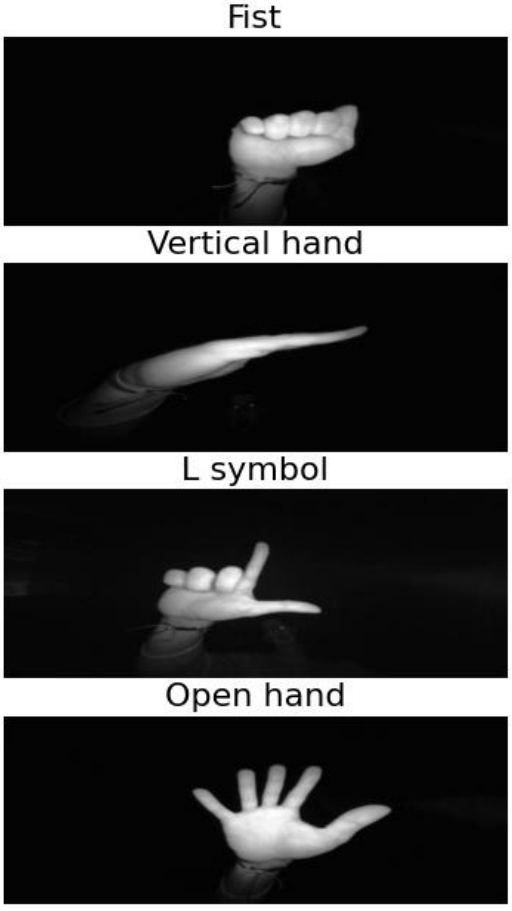


Fig. 3. Example of each hand gesture in our database.

 ISBN 978-86-7466-930-3

C. Model Architecture

VGG16 is a deep convolutional neural network pre-trained on the ImageNet dataset. The VGG16 Architecture was developed and introduced by Karen Simonyan and Andrew Zisserman. The VGG16 model achieved 92.7% accuracy on the classification task, which earned its authors second place in the competition. On the localization task, VGG16 earned first place. This model is widely used both because it is easy to implement and because it is still extremely competitive.

During training, the input to the CNN is a 224 x 224 RGB image. Subtracting the mean RGB value computed on the training set from each pixel is the only pre-processing done here. The image is passed through a stack of convolutional layers, where filters with a small radius are used (Fig. 4). The convolution stride and the spatial padding of convolutional layer input is fixed to 1 pixel. This ensures that the spatial resolution is preserved after the convolution. Five max-pooling layers, which follow some of the convolutional layers, help in spatial pooling. Max-pooling is performed over a 2×2-pixel window, with stride 2. Here is the architecture:

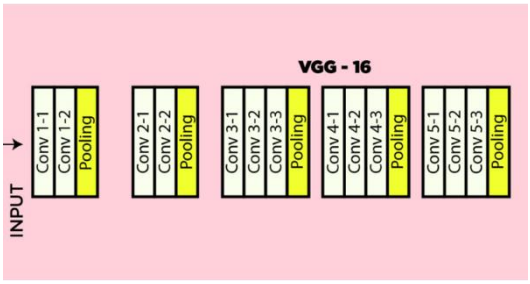


Fig. 4. VGG16 architecture, from [13].

On top of the CNN, we put a classification head that consisted of 2 dense layers. The first one was size 512 with ReLU activation function, and the second one was size 4 with Sigmoid activation function.

D. Hyper-parameters

In this section we will mention some hyper parameters as well as optimizers we used. Firstly, we used an RMS prop optimizer. This is a gradient-based optimizer that deals with vanishing and exploding gradients very well. It uses the moving average of squared gradients to normalize the gradient. This normalization decreases steps for large gradients and increases steps for small gradients. Effectively this means that the learning rate is adjusted based on previous magnitudes of gradients. For the loss function we opted for cross entropy loss.

E. Training

We trained our model during 5 epochs, with batch size 64 and learning rate 0.0001. Train-validation split ratio is 80:20. Another way to stop overfitting of the network was using regularization with L1 and L2 losses combined. Also, we used dropout layer in classification head.

IV. Results

Results of the network which was trained on the small database [11] (the one that did not contain various positions and scales of the hand), has considerably lower accuracy on the versatile test dataset. This can be seen in the confusion matrix below (Fig. 5). The accuracy of this model is 91.5%.

After training our network for five epochs on 20000 images (our database), loss reaches saturation both on validation and training set.

Further, this model has 99.3% accuracy on test data. The confusion matrix is very concentrated around the main diagonal (Fig. 6).

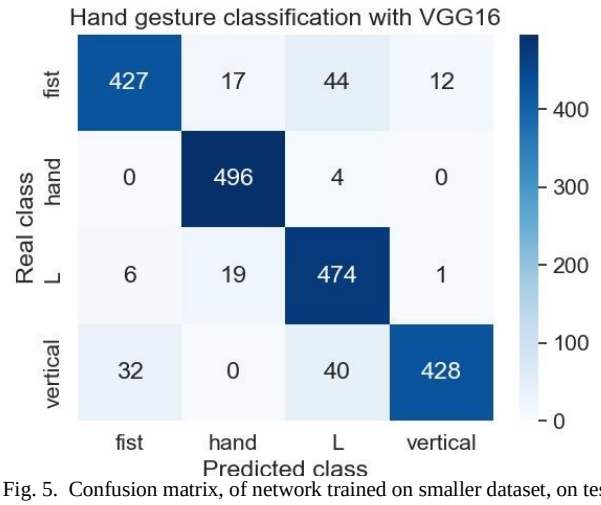


Fig. 5. Confusion matrix, of network trained on smaller dataset, on test data

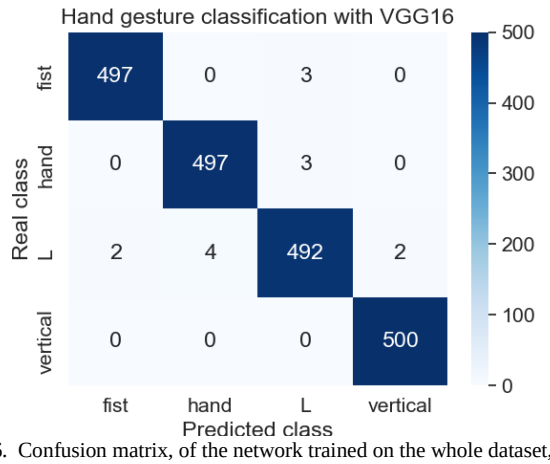


Fig. 6. Confusion matrix, of the network trained on the whole dataset, on test data

We can see that there is no significant difference in these precisions, which suggests that our model is not biased towards any class. This is because our dataset is balanced, and the network can learn the features of each class equally well. However, the L symbol has the highest number of misclassified examples. This could be because it is visually somewhere between the fist and the open hand. Two fingers extended like in the open hand, and three fingers flexed like in the fist

Next, we will show some misclassified instances:

- Fist mistaken for L shape: This could be because of

 ISBN 978-86-7466-930-3

the angle between the forearm and the fist. Therefore, this calls for a more well-rounded database that will contain lots of these examples in the training

- L shape is mistaken for fist: This could be because the index finger is too thin in the image.

In the end, it is important to notice that our system works in real-time. Also, the accuracies are even greater in real-life use because the decision is made using a majority vote from three shots of the subject's hand.

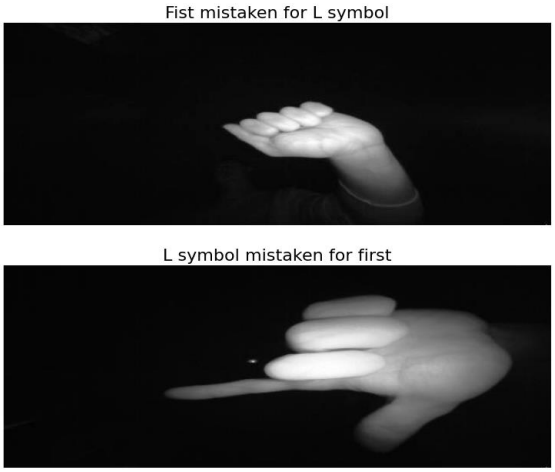


Fig. 7. Examples of failure cases.

V. Conclusion

In this paper, we developed a system for natural and non-invasive human-machine interface using a leap motion sensor. The aim was to classify four hand gesture commands with high precision in real-time. This was done by using deep convolutional network VGG16 and a custom classification head. The inference time per image is 0.23s. To trigger a decision, a hand must be detected by the leap sensor. After that, our system takes three shots at 0.33-second intervals. This setup enables us to use this system in real-time efficiently. The decision is made based on the majority vote of those three shots. This approach gives an accuracy of 99.3% on the test set. Further, our system is robust in terms of distance from the hand to the sensor, and orientation of the hand with respect to the sensor. This is important because in real-life situations workers will approach this device from different angles and heights.

Acknowledgment

This paper was funded by cascading project BrainWatch within Horizon 2020 project Human-Centered Robotics for Connected Factories - SHOP4CF (H2020 grant agreement #873087).

References

[1] Faccio, M., Bottin, M. & Rosati, G. Collaborative and traditional robotic assembly: a comparison model. Int J Adv Manuf Technol 102, 1355–1372 (2019). https://doi.org/10.1007/s00170-018-03247-z

[2] Norberto Pires, J., Godinho, T. and Ferreira, P. (2004), "CAD interface for automatic robot welding programming", Industrial Robot, Vol. 31 No. 1, pp. 71-76. https://doi.org/10.1108/01439910410512028

[3] Bachmann, D., Weichert, F. and Rinkenauer, G. 2015. Evaluation of the leap motion controller as a new contact-free pointing device. Sensors 15, 1, 214

[4] T. Gaspar et al., "Rapid hardware and software reconfiguration in a robotic workcell," 2017 18th International Conference on Advanced Robotics (ICAR), 2017, pp. 229-236, doi: 10.1109/ICAR.2017.8023523.

[5] Marin, G., Dominio, F., and Zanuttigh, P. 2015. Hand gesture recognition with jointlz calibrated leap motion and depth sensor. Multimedia Tools and Applications 1-25.

[6] Shao, L., Hand movement and gesture recognition using Leap Motion Controller.

[7] Sumpeno, S., Dharmayasa, G., Nugroho, S., Purwitasari, D., 2019. Immersive Hand Gesture for Virtual Museum using Leap Motion Sensor Based on K-Nearest Neighbor.

[8] Mutalib, M., Mohd, N., Tomari, M., Sari, S., Ambar, R.,2020. Flying Drone Controller bz Hand Gesture Using Leap Motion.

[9] Xiao, Y., Peng, Q., University of Manitoba, Canada 2017. A hand gesture-based interface for design review using leap motion controller.

[10] McCartney, R., Yuan, J., Bischof, H., Gesture Recognition with Leap Motion Controller.

[11] Lupinetti, K., Ranieri, A., Gianinni, F., Monti, M., 3D dynamic hand gestures recognition using Leap Motion sensor and convolutional neural networks.

[12] https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf , accessed 17.05.2022.

[13] https://www.geeksforgeeks.org/vgg-16-cnn-model/, accessed 17.05.2022.

[14] https://www.researchgate.net/figure/Schematic-view-of-leap-motion-controller-LMC_fig1_266614710, accessed 17.05.2022.

[15] https://www.kaggle.com/gti-upm/leapgestrecog, accessed 17.05.2022.

 ISBN 978-86-7466-930-3

Pozicioniranje Hvataljke ABB Kolaborativnog Robota Pomoću Kamere

Vojislav Vujičić, Ivan Milićević

***Abstrakt*—U ovom radu biće opisan jedan od načina pozicioniranja kolaborativnog robota pomoću kamere integrisane u hvataljku robota. U prvom delu rada biće opisana laboratorijska postavka instalirana u Naučno tehnološkom parku u Čačku. Zatim će biti opisano kreiranje virtuelne laboratorije u okviru softvera ABB Robot Studio, kao i definisanje osnovnih delova programa. Nakon kreiranja putanja robota detaljno će biti opisana opcija *Integrated vision* kao i njena implementacija pri pozicioniranju robota.**

***Ključne reč i*— Kolaborativna robotika, pozicioniranje, integrisana vizija, ABB…**

I. Uvod

Roboti su brzo poslali sastavni deo proizvodne industrije. Roboti se koriste na različite načine u različitim industrijama i imaju izuzetno veliki uticaj kada je u pitanju industrijska automatizacija.

Tradicionalni industrijski roboti su fizički odvojeni od ljudi, izvršavaju zadatke koji su odvojeni od ljudi zaposlenih u proizvodnji, kako bi obezbedili njihovu bezbednost. Sa druge strane, kolaborativni roboti su napravljeni tako da imaju ograničene brzine, i momente, kao is senzore koji im omogućavaju da budu bezbedni za ljude i da ljudi mogu biti u blizini, kao i da rade sa njima zajedno. [1,2]

Kolaborativni roboti nude nove potencijale za poboljšanje načina na koji radimo. Oni su se brzo razvili ubrzanim uvođenjem industrije 4.0 i razvojem novih robotskih tehnologija. Pre samo deset godina, prema robotima su se odnosili sa nepoverenjem, a danas je oblast industrijske robotike ona koja se najbrže razvija. Kolaborativni roboti brzo postali sastavni deo proizvodne industrije.

Ovim robotima je potrebna ljudska pomoć kada vrše manipulaciju materijala ili sklapanje nekih komplikovanih delova. Kolaborativni roboti su dizajnirani da rade i pomažu ljudima, umesto da im oduzimaju posao. Oni su programirani da obavljaju ponovljive zadatke tamo gde su ljudi potrebni, ali ne i njihova stručnost. Dakle, koriste se za jednostavne, ponavljajuće zadatke koji su opasni ili teški za ljude, kao što su operacije na proizvodnoj liniji, održavanje mašina, elektro lučno zavarivanje, rukovanje opasnim materijalima, laboratorijsko testiranje i drugi proizvodni zadaci.

Vojislav Vujičić, Fakultet tehničkih nauka u Čačku, Univerzitet u Kragujevcu, Svetog Save 65, 32000 Čačak, Srbija (e-mail: vojislav.vujicic@ftn.kg.ac.rs) (https://orcid.org/0000-0002-7037-3545)

Ivan Milićević, Fakultet tehničkih nauka u Čačku, Univerzitet u Kragujevcu, Svetog Save 65, 32000 Čačak, Srbija (e-mail: ivan.milicevic@ftn.kg.ac.rs) (https://orcid.org/ 0000-0003-0476-4991)

.

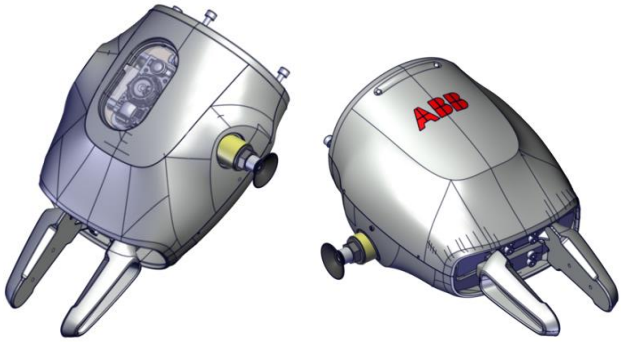
II. Laboratorijska Postavka

U ovom istraživanju je korišćena oprema postavljena u laboratoriji Naučno tehnološkog parka u Čačku čije opremanje je finansirano od strane Ministarstva prosvete , nauke i tehnološkog razvoja Republike Srbije. U ovoj laboratoriji instalirana su dve robotske ćelije. Jedna ćelija je industrijski robot ABB IRB 120, multifunkcionalni robot sa šest stepeni slobode, koji je detaljno opisan u radu [3]. Drugi robot je ABB IRB 14000 YuMi kolaborativni robot koji je prikazan na slici 1.

Slika 1. ABB IRB 14000 kolaborativni robot [4]

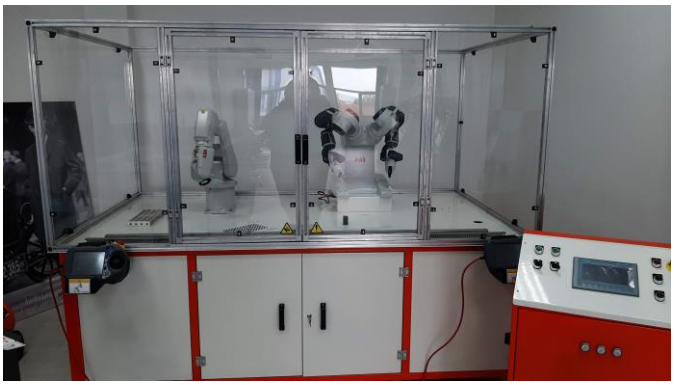
Ovaj robot je dvoručni robot koji je napravljen za zajednički rad sa čovekom na istom poslu. Robot poseduje sedam stepeni slobode u jednoj ruci koje mu daju veliku fleksibilnost i agilnost u proizvodnji. Ponovljivost robota je 0.02mm. Robot ima domet od 559mm i nosivost od 500g po ruci. IRB 14000 poseduje Smart Gripper kao end efektor sa manipulaciju i sklapanje. Griper ima servo prste kao podrazumevani modul, dok su vakuum hvataljka i kamera opcioni. U ovom slučaju leva ruka ima servo prste i vakuum modul. Desna ruka poseduje servo, vakuum i kameru (Cognex AE3) [4].

 ISBN 978-86-7466-930-3



Slika 2. Desni i levi Smart Gripper [4]

Oba robota su smeštena na radni sto sa zaštitnom zonom od pleksiglasa. Na radnom stolu nalaze aluminijumski profili za montažu dodatne opreme, senzora, pribora… Kontroleri robota i ostale komponente povezane su na komandni pult u kome se nalazi PLC kontroler i HMI tač panel. Na ovoj laboratorijskoj postavci sa opremom koja se na njoj nalazi moguće je testirati različite industrijske aplikacije. Izgled laboratorijske postavke prikazan je na slici 3.

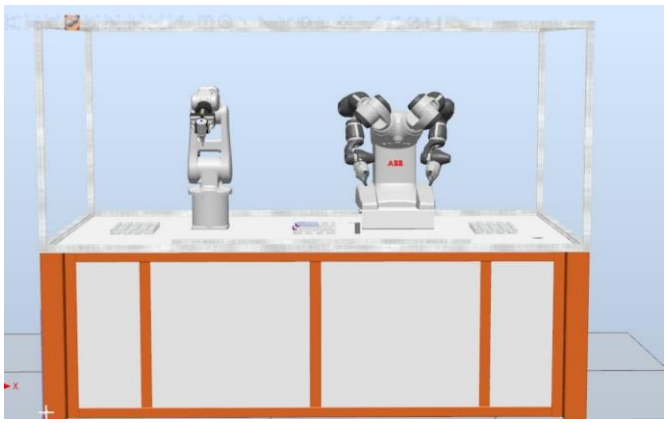


Slika 3. Izgled laboratorijske postavke

III. Virtuelno Okruženje

Jedan način za programiranje ABB robota je korišćenje softverske platforme ABB RobotStudio. Softver poseduje virtuelni kontroler za programiranje robota. Dobar je za simulaciju rada robota kao i drugih pokretnih mehanizama. Ovaj softver omogućava programiranje robota i promene programa bez ometanja ili zaustavljanja proizvodnje. RobotStudio se koristi kao virtuelna laboratorija i programiranje robota van mreže (offline). Ova softverska platforma sastoji se od četiri glavna dela: modeliranje, simulaciju, kontroler i RAPID [5, 6].

Modeliranje omogućava kreiranje virtuelnog okruženja izradom ili uvozom 3D modela mehaničkih delova i mašina u proizvodnoj liniji. Virtuelno okruženje je korisno za izbegavanje mehaničkih sudara između robota i drugih delova proizvodne linije. Kreiranje virtuelne laboratorije počinje kreiranjem nove prazne stanice. Sledeći korak je uvoz mehaničkog 3D modela radnog prostora sa sigurnosnom zonom iz softvera za 3D modeliranje. Ovaj korak je detaljno opisan u [3]. Nakon dodavanja radnog stola, robot mora biti postavljen na radni sto. ABB IRB 14000 je dodat iz biblioteke. Položaj robota je podešen (pozicija (x, y, z): 1604, 900, 853, orijentacija (x, y, z): 0,0, -90). Sledeći korak je dodavanje Smart Gripper-a na levu i desnu ruku. Leva ruka ima servo prste i vakuum, dok desna ima servo prste, vakuum i kameru. Hvataljke moraju biti pričvršćene za svaku ruku. Nakon postavljanja hvataljki potrebno je dodati 3D model radnog komada da bi se kompletirao mehanički model virtuelne laboratorije. Kompletan mehanički virtuelni model je prikazan na slici 4.



Slika 4. Izgled virtuelne laboratorije [3]

Nakon dodavanja svih potrebnih mehaničkih delova u virtuelni model, virtuelni kontroler se mora dodati. Dodat je kontroler sa RobotWare verzija 6.11.01 za IRB 14000 0,5kg 0,5m.

IV. Programiranje Kretanja Robota

Program koji će biti napravljen je da se sa jedne (desne) strane podigne predmet sa jednom (desnom) rukom, prebaci u drugu (levu) ruku i spusti na drugu stranu radnog prostora. Pre preuzimanja predmeta sa desnom rukom, kamera na hvataljci skenira prostor i vrši orjentaciju hvataljke u zavisnosti od orjentacije predmeta koji se prenosi.

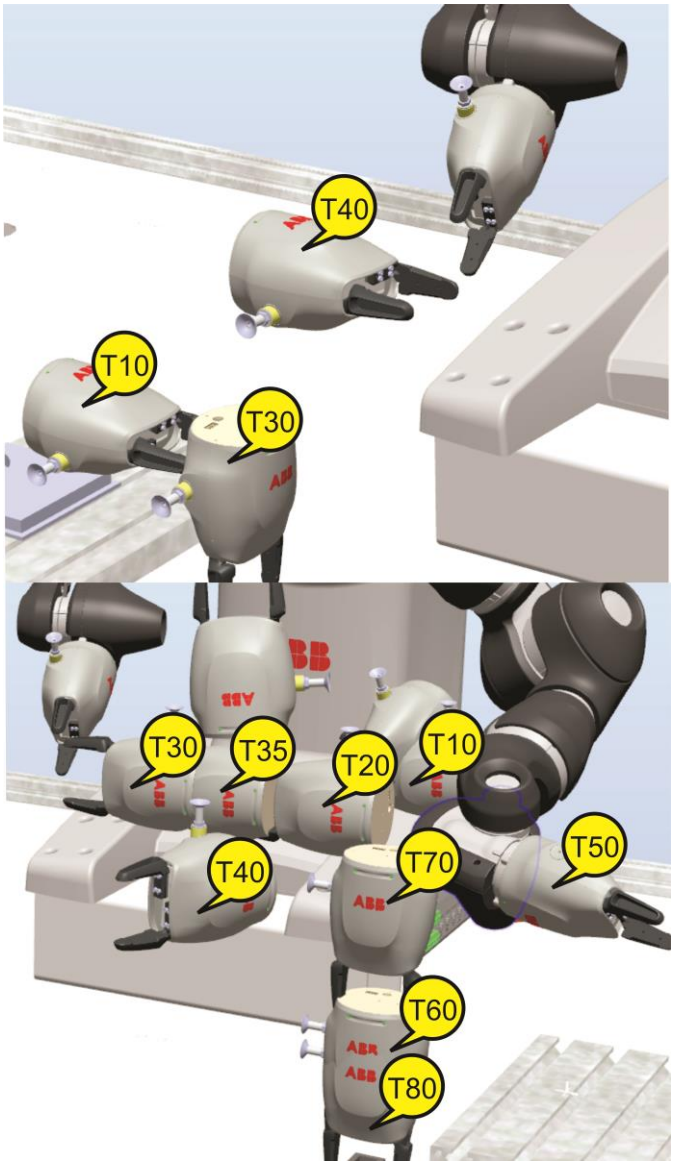
Programiranje ABB robota počinje definisanjem *Target-a*. *Target* je specifična tačka u radnoj zoni robota u koju robot treba da dođe. Jedan od najbitnijih *Target-a* je *Target* u inicijalnom (*Home)* položaju ruke. Za svaku ruku potrebno je podesiti inicijalnu poziciju. Svaka ruka ima svoje *Target-e*

Desna ruka mora da ode iz početnog položaja u položaj iznad predmeta, poziciju za snimanje kamerom, Target 10. Na poziciji Targeta 30, hvataljka mora biti zatvorena da bi se uhvatio predmet. Zatim ide do Targeta za razmenu – Target 40. Slika 5a prikazuje desnu poziciju Smart Gripper-a u definisanim Targetima. Od tih Targeta se mora napraviti putanja (Path_R). Ova putanja sadrži sve napravljene Targete i instrukcije za pomeranje *Move* (*J-joint or L-linear*) za prelazak sa jednog na drugi Target.

Leva ruka ima *Target-e* za približavanje tački razmene desnom rukom (Target 10, 20, 30), izvlačenje iz tačke razmene (Target 35), Target (40, 50, 60 i 80) služe za prebacivanje predmeta na drugu poziciju na radnom prostoru. Poslednji Target 70 je pomeranje od Targeta 80 da bi se

 ISBN 978-86-7466-930-3

izbegao sudar hvataljke sa predmetom. Slika 5b prikazuje položaj levog Smart Gripper-a u definisanim Targetima. Sa tim Targetima, napravljena je putanja do (Path_L).



Slika 5a. Desna hvataljka u definisanim Targetima, Slika 5b. Leva hvataljka u definisanim Targetima

Nakon kreiranja programa, moguće je izvršiti simulaciju rada. Kada simulacija počne, robot ide od Targeta do Targeta, praveći pokrete definisane putanjama. Prva simulacija pokazuje da ruke nisu sinhronizovane, pa se program mora modifikovati. Promene koje se moraju izvršiti su:

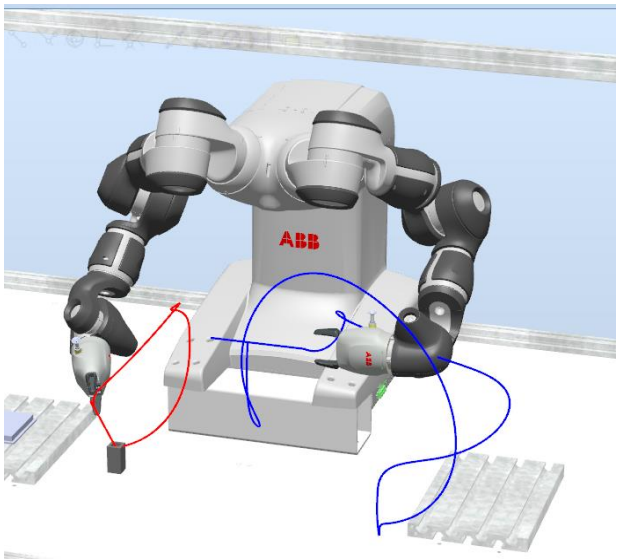
a) Desna ruka mora da sačeka dok leva ruka ne preuzme predmet.
b) Leva ruka mora da sačeka dok se desna ruka ne pomeri iz zone hvatanja.
c) Desna ruka može ići u početni položaj, a leva ruka može da nastavi da prenosi predmet na željenu pozuciju.
d) Sačekati da obe ruke budu u početnoj poziciji.

Program mora imati promeljive za sinhronizaciju ruku:

```
PERS tasks task_list{2}:=[["T_ROB_L"],["T_ROB_R"]];
VAR syncident sy01; VAR syncident sy02; VAR syncident sy03;
```

Definisane promeljive se koriste u fukncji za sinronizaciju WaitSyncTask\InPos,sy01,task_list; i dodaju se u program za obe ruke na mestu gde je potrebno da ruke jedna drugu čekaju.

Nakon dodavanja delova za sinhronizaciju simulacija kretanja robota sa putanjom kretanja ove ruke data je na slici 6.



Slika 6. Simulacija kretanja robota sa prikazanim putanjama kretanja

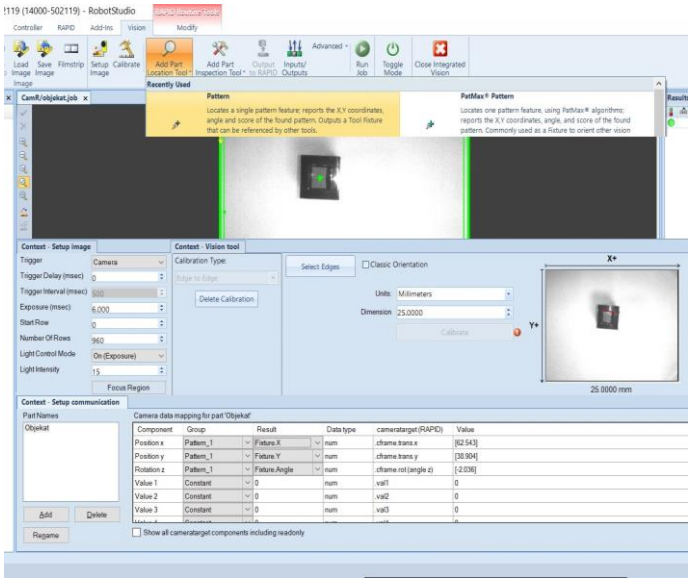
V. Podešavanje Vizijskog Sistema

Za pozicioniranje robota na osnovu vizijskog sistema koristi se desna ruka robota koja u sebi ima integrisanu kameru (Cognex AE3). Pre podešavanja opcije *Inegrated vision* potrebno je desnu ruku robota postaviti u poziciju za snimanje objekta i modifikovati Target_30. Nakon toga se može odabrati opcija *Integrated vision* u kojoj je potrebno izvršiti sledeće korake:

1. Opcija *Setup Image* gde se podešavaju parametri ekspozicije i osvetljaja podloge.
2. Opcija *Calibrate* kojom se definiše koliko piksela na slici je dimenzija predmeta (predmet je dimenzije 25x25 mm).
3. Opcija *ADD Part Location Tool/Patern* kojom se definiše oblik predmeta kome treba podesiti poziciju.
4. Opcija *Output to RAPID* gde se definiše šta će sa slike RAPID program preuzeti od podataka.
5. Opcija *Save Job* kojom se definisani parametri snimaju na kontroleru robota.

Definisanje parametara u okviru *Integrated vision-a* dat je prikazano je na slici 7.

 ISBN 978-86-7466-930-3



Slika 7 Podešavanje vizijskog sistema

Po završetku definisanja parametara u okviru *Integrated vision* potrebno je u RAPID programu napisati kod kojim se poziva rad kamere i vrši upis pozicije hvatanja predmeta na osnovu slike. Deo koda koji vrši pozicioniranje na osnovu slike je:

```
CamSetProgramMode CamR;
CamLoadJob CamR, Kamera;
CamSetRunMode CamR;
CamReqImage CamR;
CamGetResult CamR, Target_pozicija_sa_kamere;
wobj0.oframe := Target_pozicija_sa_kamere.cframe;
```

Nakon implementacije programa izvršeno je testiranje programa, finalno podešavanje pozicija za razmenu predmeta, robot je pušten u rad. Robot u radu prikazan je na slici 8

Slika 8. Robot ABB IRB 14000 YuMi u radu

VI. ZAKLJUČAK

Opremanjem nove laboratorije opremljene sa dva ABB robota u okviru Naučno-tehnološkog parka u Čačku otvorilo je nove mogućnosti za razvoj robotike na Fakultetu tehničkih nauka u Čačku. Virtuelna laboratorija sa ABB IRB 120 i ABB IRB 14000 je formirana korišćenjem softvera RobotStudio. Ovo omogućava programiranje robota od kuće koristeći virtuelno okruženje, kao i primenu u izvođenju laboratorijskih vežbi sa studentima.

Program za kretanje robota napravljen je u virtuelnoj laboratorij nakon čega je proveren i podešen kroz simulaciju. Nakon toga je program ubačen u kontroler robota i na realnom robotu izvršena finalna podešavanja.

Nakon definisanja kretanja u program je ubačen vizijski sistem kojim se vrši pozicioniranje hvataljke robota bez obzira na poziciju i orjentaciju predmeta na podlozi.

Naredni koraci u istraživanju oblasti vizije i kolaborativne robotike su primena ovakvih sistema u radu sa čovekom, gde bi na osnovu slike robot mogao da odluči šta je naredno potrebno uraditi čime se postiže veća fleksibilnost.

Takođe je moguće koristiti viziju za inspekciju urađenog zadatka od strane čoveka ili nekog drugog robota i korigovanje u koliko je zadatak moguće ispratiti.

ZAHVALNICA

Istraživanja prezentovana u ovom radu su delimično finansirana sredstvima Ministarstva prosvete, nauke i tehnološkog razvoja RS, ugovor br. 451-03-68/2022-14/200132 čiji je realizator Fakultet tehničkih nauka u Čačku - Univerziteta u Kragujevcu.

LITERATURA

[1] Vicente, L. i ostali „Industrial Collaborative Robotics Platform. In: Camarinha-Matos, L.M., Boucher, X., Afsarmanesh, H. (eds) Smart and Sustainable Collaborative Networks 4.0. PRO-VE 2021.“ IFIP Advances in Information and Communication Technology, vol 629. Springer, Cham. https://doi.org/10.1007/978-3-030-85969-5_53

[2] Matjaž, M. i ostali. “Robotics” Springer Cham 2019. https://doi.org/10.1007/978-3-319-72911-4

[3] Vujičić, V., i ostali, „Offline Robot Programming Using ABB RobotStudio“, X International Scientific Conference Heavy Machinery HM 2021, Proceedings HM 2021, ISBN: 978-86-81412-09-1, pp. C79-C83, University of Kragujevac, Faculty of Mechanical and Civil Engineering Kraljevo, 23-25th June 2021, Vrnjačka Banja, Serbia.

[4] Product specification IRB 14000 [Online]. Available: https://search.abb.com/library/Download.aspx?DocumentID=3HAC052982-001&LanguageCode=en&DocumentPartId=&Action=Launch [Pristup 22.4.2022.]

[5] Podobnik. J., i ostali, „Osnove robotike: Laboratorijski praktikum“ Založba FE, Universa v Ljubljani, Ljubljana 2018,

[6] Adit, M., „HandBook Guidance on the programming of ABB YuMi IRB 14000“ Otto-von-Guericke-University Magdeburg 2020. https://doi.org/10.13140/RG.2.2.12746.18881

Upravljanje pasivnom krutošću završnog uređaja robota oblikovanjem elipsoida krutosti

Branko Lukić, Nikola Knežević i Kosta Jovanović

***Apstrakt*— Krutost završnog uređaja robota određuje ponašanje robota pri interakciji sa okolinom. Krutost završnog uređaja je najčešće predstavljena preko matrice krutosti čije oblikovanje može da bude neizvodljivo jer roboti ne poseduju dovoljno stepeni slobode da bi se svi elementi u matrice krutosti podesili. To dovodi do primene optimizacije koja balansira između vrednosti elemenata matrice krutosti koji su od interesa za izvršavanje zadatka. U ovom radu je predložen pristup za „*offline*“ oblikovanje matrice krutosti primenom elipsoida krutosti gde se podešavanjem orijentacije i dužina osa elipsoida oblikuje krutost. Oblikovanje elipsoida ima manje parametara koje je potrebno podesiti u odnosu na matricu krutosti. Predložene su dve kriterijumske funkcije za oblikovanje elipsoida krutost koje eksploatišu kinematičku redundansu robota. Optimalne vrednosti pozicija i krutosti zglobova izračunate su primenom algoritma zasnovanog na SLSQP (engl. *Sequential Least Square Programming*). Nakon dobijanja željenog oblika elipsoida krutosti završnog uređaja robota, pokazana je mogućnost promene volumena elipsoida na željenu vrednost skaliranjem krutosti u zglobovima.**

***Ključne reči*—Elipsoid krutosti, krutost završnog uređaja, pasivna krutost.**

I. Uvod

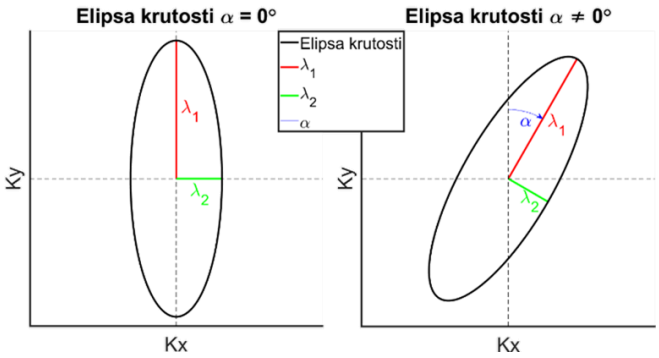
Kod bezbedne fizičke interakcije između čoveka i robota (pHRI) [1, 2] ili robota i okoline, jednu od bitnijih uloga igra popustljivost robota. Pored aktivne popustljivosti [3, 4] koja se ostvaruje kroz upravljačku petlju, postoji i pasivna popustljivost koja se realizuje kroz elastične elemente ugrađene u prenosnom sistemu robota između aktuatora i segmenta robota [5]. Fokus ovog rada je na robote pogonjene aktuatorima sa promenljivom krutošću – VSA (engl. *Variable Stiffness Actuators*), koji imaju dve kontrolne varijable: pozicije i krutosti aktuatora.

Matrica krutosti završnog uređaja je u funkciji pozicija zglobova robota preko Jakobijeve matrice i krutosti zglobova (videti (4) i (5)). To znači da za kinematički redundantnog robota sa popustljivim aktuatorima, krutost završnog uređaja se može oblikovati promenom konfiguracije u nultom prostoru i/ili promenom krutosti zglobova. Za tipične robotske konfiguracije, matrica krutosti

Branko Lukić – Elektrotehnički fakultet, Univerzitet u Beogradu, Bulevar Kralja Aleksandra 73, 11020 Beograd, Srbija (e-mail: branko@etf.rs).

Nikola Knežević – Elektrotehnički fakultet, Univerzitet u Beogradu, Bulevar Kralja Aleksandra 73, 11020 Beograd, Srbija (e-mail: knezevic@etf.rs).

Kosta Jovanović – Elektrotehnički fakultet, Univerzitet u Beogradu, Bulevar Kralja Aleksandra 73, 11020 Beograd, Srbija (e-mail: kostaj@etf.rs).



Sl. 1. Elipsa krutosti je definisana dužinama osa λ_1.i λ_2, kao i uglom rotacije α. Levo: Ose elipse krutosti se poklapaju sa osama koordinatnog sistema. Desno: Pravac u kome je elipsa krutosti usmerena je zarotiran za ugao α.

je simetrična matrica dimenzija 6×6, što rezultuje sa 21 promenljivom. Kontrola matrice krutosti i svih šest pozicija završnog uređaja (3 rotacije i 3 translacije) [6, 7] rezultuje sa 27 promenljivih koje treba podesiti.), što je sa stanovišta kontrole veoma zahtevno budući da postoji suviše promenljivih i fizičkih ograničenja (ograničen raspon pokreta zglobova i krutosti zglobova) i manjak kontrolnih ulaza. Za kontrolu svih 27 promenljivih veličina robota pogonjenog sa aktuatorima promenljive krutosti minimalan broj aktuatora bi bio 14 [6], ne razmatrajući fizička ograničenja koja postoje i da bi realan broj bio daleko veći i kompleksniji za realizaciju i upravljanje. Stoga, kontrola cele matrice krutosti nije laka, niti je potpuno opravdan cilj [8], već da je kontrola dijagonalnih elemenata matrice krutosti neophodna u većini aplikacija.

Matrica krutosti završnog uređaja ima nelinearnu zavisnost od konfiguracije robota preko Jakobijeve matrice, pa za tipičnog robota sa 6-7 stepeni slobode, nemoguće je pronaći analitički skup pozicija zglobova koji će zadovoljiti željeni položaj završnog uređaja i matricu krutosti. Stoga su neophodne tehnike optimizacije koje koriste nulti prostor [4, 9, 10].

Oblikovanje krutosti završnog uređaja robota računanjem optimalnog seta krutosti u zglobovima za konstantnu poziciju završnog uređaja, gde su krutosti zglobova predstavljen kroz linearni sistem jednačina, a gde je linearnost ostvarena izborom odgovarajuće norme dato je u [6]. Nadogradnja računanja optimalnog seta krutosti u zglobovima prikazana je [7]. U njemu se razmatra iterativna procedura za rešavanje nelinearnih sistema jednačina sa ograničenjima (radni opsezi pozicija i krutosti zglobova). Pristup optimizaciji kretanjem u nultom prostoru može

 ISBN 978-86-7466-930-3

dovesti do lokalno optimalnog rešenja. Kada se računa optimalna konfiguracija zgloba, projekcija nultog prostora će osigurati glatku putanju zgloba u poređenju sa metodama nelinearne optimizacije koji traže optimalno globalno rešenje koje se može značajno razlikovati od trenutne poze robota [11].

Pristup predložen u ovom radu se zasniva na „*offline*“ oblikovanju elipsoida krutosti u fazi planiranja trajektorije i krutosti završnog uređaja robota, čijim se oblikovanjem indirektno podešava i matrica krutosti završnog uređaja. Predložene su dve kriterijumske funkcije. Jedna koja ima labaviji kriterijum i jedna sa konzervativnijim kriterijumom.

Struktura rada je sledeća: Nakon uvodnog poglavlja sledi poglavlje II gde je modelirana i opisana priroda matrice krutosti. Poglavlje III opisuje grafičku prezentaciju krutosti preko elipsi i elipsoida krutosti. U poglavlju IV je opisan algoritam za offline planiranje krutosti i pozicije završnog uređaja robota sa simulacionim rezultatima, a u poglavlje V je diskusija i zaključak.

II. PASIVNA KRUTOST

Lokalna statička karakteristika krutosti zglobova robota se definiše kao negativni izvod momenta po uglu [6, 7]

$$K_j = -\frac{\partial \tau}{\partial q}, \tag{1}$$

gde je K_j matrica krutosti zglobova, τ je vektor momenata koji se razvijaju na zglobovima, a q je vektor pozicija zglobova. Važi relacija $\tau = J(q)^T F_{ekst}$, između Jakobijeve matrice ($J(q)$), eksterne sile koja deluje na završni uređaj robota (F_{ekst}) i momenta zglobovima (τ) koji se stvara usled delovanja sile, kao i to da je F_{ekst} proporcionalno matrici krutosti završnog uređaja K_C i otklona iz ravnotežnog položaja Δx nastalog usled eksterne sile $F_{ekst} = K_C \Delta x$, tako da izraz (1) postaje

$$K_j = -\frac{\partial (J(q)^T K_C \Delta x)}{\partial q} = J(q)^T K_C J(q) - \frac{\partial J(q)^T}{\partial q} K_C \Delta x, \tag{2}$$

gde je $\partial \Delta x / \partial q = J(q)$, K_C je $m \times m$ simetrična matrica, K_j je $n \times n$ dijagonalne matrice, $J(q)$ je $m \times n$ matrica i q je n-dimenzionalni vektor pozicija zglobova. Parametri m i n su dimenzije radnog prostora i broja zglobova robota. Kada nema otklona od ravnotežnog stanja, tada je $\Delta x = 0$, tako da izraz (2) postaje

$$K_j = J(q)^T K_C J(q). \tag{3}$$

Matrica pasivne krutosti zglobova (K_j) može da ima nedijagonalnu strukturu sa van dijagonalnim elementima kada se koriste biartikulisani aktuatori [12, 13], gde jedan aktuator pogoni više od jednog segmenta robota, a koji su mehanički veoma komplikovani za realizaciju. Fokus u ovom radu je na uniartikulisanim aktuatorima koji rezultuju da matrica K_j ima dijagonalnu formu.

Iz jednačine (3) se dobija statička relacija za mapiranje matrice krutosti završnog uređaja robota K_C u Dekartovim koordinatama (engl. *Cartesian space*) sa jedne strane i sa druge strane matrice krutosti zglobova K_j i konfiguraciji robota predstavljene kroz Jakobijevu matricu $J(q)$ [6, 7] kao

$$K_C = \left(J(q) K_j^{-1} J(q)^T\right)^{-1}, \tag{4}$$

odnosno

$$K_C = J(q)^{\dagger^T} K_j J(q)^{\dagger}, \tag{5}$$

gde je $J(q)^{\dagger}$ desna pseudo inverzija Jakobijeve matrice $J(q)$ za koju važi $J(q)J(q)^{\dagger} = I$, gde je I jedinična matrica. U opštem slučaju pseudo inverzija je definisana kao $J(q)^{\dagger} = Z^{-1}J(q)^{\mathrm{T}}(J(q)Z^{-1}J(q)^{\mathrm{T}})^{-1}$, gde je Z pozitivno definitivna matrica. Za pravilno mapiranje krutosti u zglobu i krutosti završnog uređaja metrički tenzor (Z) mora imati vrednost matrice krutosti zglobova $Z = K_j$ [6].

Jednačine (4) i (5) ukazuju da se na krutost i popustljivost završnog uređaja robota može uticati:

1) rekonfiguracijom robota kroz nulti prostor kinematički redundantnog robota,
2) promenom krutosti zglobova robota pogonjenih sa VSA,
3) simultanom rekonfiguracijom i promenom krutosti zglobova robota.

Svi nabrojani načini se mogu primeniti na robote pogonjene sa aktuatorima promenljive krutosti, dok samo rekonfiguracija robota kroz nulti prostor se može primeniti na robote pogonjene aktuatorima sa konstantnom krutošću. Zbog toga eksploatacija redundanse kod robota pogonjenog aktuatorima konstantne krutosti ima veliku važnost.

Česta aproksimacija je da se matrica krutosti podeli na četiri submatrice (K_T i K_R su translatorna i rotaciona krutost, dok su K_{RT} i K_{TR} matrice kuplovanja), gde su od interesa samo submatrice na glavnoj dijagonali (translatorna i rotaciona), dok se one na sporednoj dijagonali zanemaruju [6]

$$K_C = \begin{bmatrix} K_T & K_{RT} \\ K_{TR} & K_R \end{bmatrix}, \tag{6}$$

$$\widehat{K}_C = \begin{bmatrix} K_T & 0 \\ 0 & K_R \end{bmatrix}. \tag{7}$$

Treba imati na umu da je $\widehat{K}_C$ aproksimacija koja je lokalno validna.

III. GRAFIČKA REPREZENTACIJA KRUTOSTI

Za robote koji imaju bilo aktivnu ili pasivnu popustljivost, moguće je sprovesti određene analize koje se tiču performansi robota i krutosti završnog uređaja [14, 15, 9]. Generalno, elipse i elipsoidi daju jednu vrstu aproksimacija koliko i u kom pravcu robot može da razvije krutost, što predstavlja koristan alat u fazi dizajniranja kao i u operativnoj fazi. Elipsoidi imaju veliku primenu u analizi i dizajnu matrica krutosti kod aktivne popustljivosti, gde se kod kinematički redundantnih robota mogu koristiti da se maksimizira radni prostor [14, 15], a da pri tome aktuatori ne budu u zasićenju i željeno ponašanje ne bude narušeno.

 ISBN 978-86-7466-930-3

Kod aktivne krutosti osim samih elipsoida i politopa moguće je uvesti i region ostvarive krutosti (SFR – engl. *Stiffness Feasibility Region*) koji predstavlja oblik sa ne-politopskim granicama [14].

Kod sila interakcije, bez obrtnih momenata, koje deluju na završni uređaj, ulogu igraju samo elementi matrice translatorne krutosti K_T koji se odnose na sile odnosno na translatorna kretanja. To je kvadratna simetrična matrica koja ima dimenzije 3×3 za trodimenzionalni prostor, ili 2×2 u ravanskom prostoru (2D). Kada je u pitanju trodimenzionalni prostor, matrica krutosti se može predstaviti pomoću elipsoida krutosti, ili u slučaju dvodimenzionalnog prostora pomoću elipse krutosti. Radi intuitivnijeg razumevanja na slici 1 je dat prikaz elipse krutosti. Parametri koji opisuju elipsu krutosti su ose elipse λ_1 i λ_2 (u slučaju elipsoida postoji i λ_3) međusobno normalne, i ugao orijentacije elipse α (u slučaju elipsoida postojaće dodatni ugao orijentacije u odnosu xy ravan). Pravac orijentacije elipse i elipsoida je u pravcu najduže ose elipse/elipsoida. Na levoj strani na slici 1 prikazana je elipsa kada se ose elipse poklapaju sa osama koordinatnog sistema, dok na desnoj strani je prikazana elipsa koja je ukošena za ugao α. Kada se ose elipse/elipsoida poklapaju sa osama koordinatnog sistema, tada nema kuplovanih elemenata i matrica krutosti ima dijagonalni oblik.

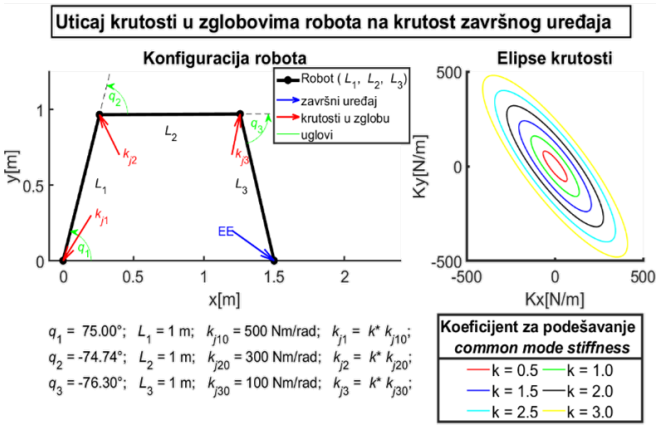
Dužine osa i ugao orijentacije elipsoida se dobija dekompozicijom matrice (SVD – engl. *Singular Value Decomposition*) krutosti kao

$$K_T = USV^T, \qquad (8)$$

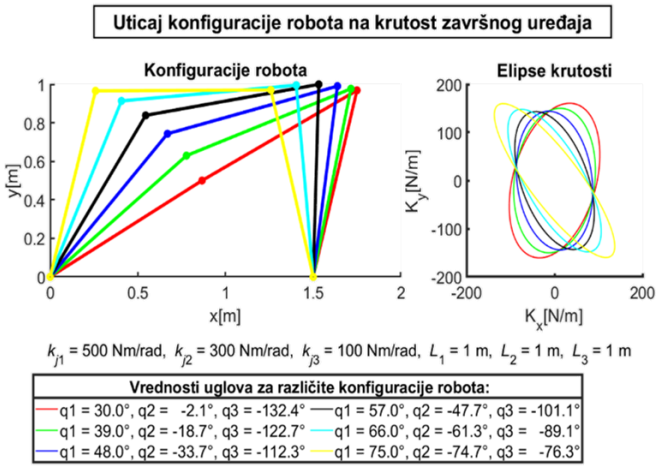
Kolone ortogonalnih matrica U i V predstavljaju sopstvene vektore, a elementi na dijagonali dijagonalne matrice S predstavljaju sopstvene vrednosti. Sopstvene vrednosti matrice S predstavljaju intenzitete krutosti duž osa elipsoida, a vrednosti kolona matice V predstavljaju sopstvene vektore koji određuju pravac orijentacije osa elipse/elipsoida.

Oblikovanje krutosti završnog uređaja može se podeliti na krutost zavisnu od konfiguracije (CDS – engl. *Configuration Dependant Stiffness*) i krutost koja zavisi od krutosti zglobova (CMS – engl. *Common Mode Stiffness*) [16]. CDS potiče od redundantne konfiguracije robota i koristi se za usmeravanje elipse ili elipsoida krutosti, dok CMS utiče na veličine osa elipse ili elipsoida. CMS predstavlja dodatni parametar koji se može uključiti u oblikovanje krutosti. Krutost završnog uređaja robota pogonjenog sa aktuatorima konstantne krutosti može se promeniti samo rekonfiguracijom kada je robot kinematički redundantan.

Na primeru planarne konfiguracije robota sa 3 stepena slobode (segmenti L_1, L_2 i L_3) koja je ilustrovana slici 2 (levo) prikazan je uticaj CMS na elipsu krutosti (slika 2 desno). Pozicije zglobova (q_1, q_2 i q_3) su konstantne i samim tim pozicija završnog uređaja (EE) je konstantna. Krutost zglobova (k_{j1}, k_{j2} i k_{j3}) dobija se kao proizvod početnih vrednosti krutosti (k_{j10}, k_{j20} i k_{j30}) i parametra CMS-a (k) kao $k_{j1} = kk_{j10}$, $k_{j2} = kk_{j20}$ i $k_{j3} = kk_{j30}$. CMS ne utiče na oblik (odnos osa) i usmerenje elipse



Sl. 2. Uticaj *Common Mode Stiffness* -a na krutost završnog uređaja robota. Levo: Planarna konfiguracija robota koja se sastoji od tri segmenta – L_1, L_2 i L_3 (crno); pozicija završnog uređaja robota (plavo); krutosti u zglobovima k_{j1}, k_{j2} i k_{j3} (crveno); pozicije zglobova q_1, q_2 i q_3 (zelena). Desno – Elipsoidi krutosti za fiksnu konfiguraciju ($q_1 = 75°$, $q_2 = -74.74°$ i $q_3 = -76.30°$) različiti koeficijent CMS-a $k \in [0.5 - 3]$. Krutost u zglobovima se računa kao proizvod početnih vrednosti krutosti (k_{j10},k_{j20} i k_{j30}) i parametra common mode stiffness-a (k) kao $k_{j1} = kk_{j10}$, $k_{j2} = kk_{j20}$ i $k_{j3} = kk_{j30}$.



Sl. 3. Uticaj konfiguracije kinematički redundantnog robota na krutost završnog uređaja robota sa dužinama segmenata $L_1 = 1m$, $L_2 = 1m$ i $L_3 = 1m$, i krutostima u zglobovima $k_{j1} = 500\text{Nm/rad}$, $k_{j2} = 300\text{Nm/rad}$ i $k_{j3} = 100\text{Nm/rad}$. Levo: Različite planarne konfiguracije robota sa istom pozicijom završnog uređaja; Desno – Elipsoidi krutosti za različite konfiguracije robota.

krutosti, već samo na njen volumen odnosno samo linearno skalira elemente matrice krutosti.

Jedan od pristupa za oblikovanje matrice ili elipse (ili elipsoida u 3D prostoru) krutosti robota pogonjenog aktuatorima konstantne krutosti je kroz promenu kinematičke konfiguracije robota kada postoji kinematička redundansa. Promena konfiguracije se odvija u nultom prostoru. To omogućava da robot pored primarnog zadatka (praćenje trajektorije) izvrši i sekundarni zadatak koji je u ovom slučaju oblikovanje krutosti završnog uređaja robota. Na primeru planarne konfiguracije robota sa 3 stepena slobode (segmenti L_1, L_2 i L_3) koja je ilustrovana slici 3 (levo) prikazan je uticaj CDS na elipsu krutosti (slika 3 desno).

 ISBN 978-86-7466-930-3

IV. ALGORITAM ZA OFFLINE PLANIRANJE KRUTOSTI I POZICIJE ZAVRŠNOG UREĐAJA ROBOTA

Osim preko matrice krutosti, krutost završnog uređaja je moguće oblikovati preko elipse krutosti. U ovom radu predložen je offline pristup oblikovanja krutosti završnog uređaja robota, promenom optimizacionih algoritama, koji je predviđen za planiranje krutosti kada je zadatak u napred poznat, pa samim tim i trajektorija završnog uređaja. Primenjena su dva kriterijuma koji kombinuju oblik i grešku orijentacije elipsoida. Oblikovanje elipsoida krutosti se ostvaruje kombinacijom nultog prostora i krutosti u zglobovima. Optimizacioni algoritmi su primenjeni na simulacionom modelu KUKA LWR robota na primeru ravanskog kretanja u „xy" ravni, kada se završni uređaj robota kreće po krivoj putanji.

Optimizacioni algoritmi koji su korišćeni su zasnovani na SLSQP (engl. Sequential Least SQuare Programming) [10, 17, 18], a implementirani su Matlab-u. SLSQP je algoritam koji optimizuje vrednosti nelinearnih kriterijumskih funkcija sa linearnim i nelinearnim ograničenjima. Ograničenja koja postoje u predloženim optimizacionim algoritmima se odnose na ostvarive vrednosti pozicije zglobova robota i na definisani opseg krutosti koje robot može da ostvaruje. Nelinearna ograničenja koja moraju biti zadovoljena se odnose na kinematiku. Optimizacioni algoritam uvek mora da zadovolji željenu poziciju završnog uređaja robota.

Kako zbog prirode zadatka, pozicija zadnjeg zgloba ne utiče na položaj završnog uređaja, već samo na ugao orijentacije oko z ose, njegova vrednost je zaključana na 0°. Na taj način robot ima 6 stepeni slobode na raspolaganju za izvršavanje zadatak. Usvojen je da raspon krutosti zglobova bude 1-100 Nm/rad. Ovo se može usvojiti bez gubljenja opštosti, jer bitan je relativan odnos krutosti u zglobovima za orijentaciju i odnos osa elipsoida krutosti. Naknadno preko CMS-a se podešava i volumen.

Prvi predloženi kriterijum (f_1) usmerava elipsoid i maksimizira krutost duž pravca kretanja, zadavajući minimalni odnos dijagonala elipsoida. Drugi kriterijum (f_2) usmerava elipsoid duž pravca kretanja, ali zadajući tačan odnos dijagonala elipsoida. Predložene optimizacione metode imaju za cilj da doprinesu planiranju trajektorije i krutosti zglobova prilikom izvršavanja zadataka.

A. *Maksimiziranje krutosti duž pravca kretanja*

Oblikovanje elipsoida krutosti moguće je ostvariti primenom kriterijumske funkcije definisane kao

$$f_1 = \frac{|\alpha_{error}| + c}{min\left(\frac{r_{max}}{r_{min}}, r\right)/r}, \tag{9}$$

gde α_{error} označava grešku orijentacije elipsoida u stepenima, r je minimalni željeni odnos osa elpsoida, r_{max} i r_{min} su najduža i druga po dužini osa elipsoida, dok je c pozitivna konstanta koja omogućava optimizaciju kada je apsolutna greška orijentacije elipsoida $|\alpha_{error}| = 0°$, dok je željena orijentacija u smeru kretanja završnog uređaja robota. Usvojena je vrednost $c = 0.01$.

U svakoj iteraciji računaju se pozicije i krutosti zglobova koje minimiziraju vrednost funkcije f_1 kao

$$\begin{bmatrix} q[p] \\ k_j[p] \end{bmatrix} = \underset{\substack{q\in[q_{min},q_{max}] \\ k_j\in[k_{j_{min}},k_{j_{max}}]}}{argmin} f_1(q, k_j), \tag{10}$$

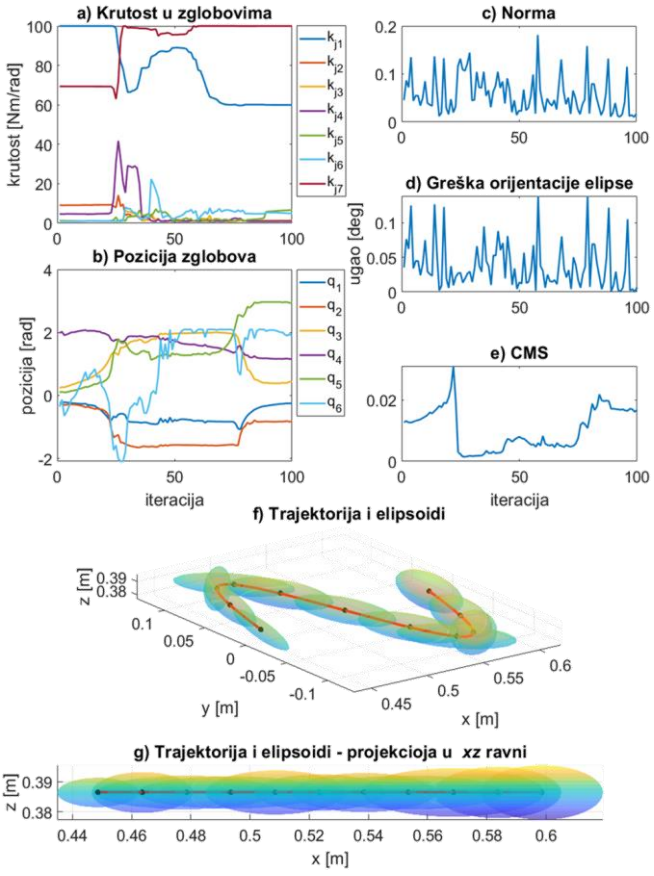
gde je p redni broj iteracije, a q_{min}, q_{max}, $k_{j_{min}}$ i $k_{j_{max}}$ su minimalne i maksimalne vrednosti pozicija i krutosti zglobova. Opseg vrednosti promenljivih u optimizacionom algoritmu se može dodatno suziti, i samim tim smanjiti prostor pretrage i ubrzati algoritam pretrage u svakoj iteraciji. Nove granične vrednosti varijabli (q^*_{min}, q^*_{max}, $k^*_{j_{min}}$ i $k^*_{j_{max}}$) u svakoj iteraciji se računaju uzimajući u obzir maksimalne brzine promene koje robot može da ostvari u okviru jedne iteracije (za pozicije zglobova - Δq, za krutost zglobova - Δk_j) i fizički ostvarive vrednosti kao

$$q_{min\_iter}[p] = max(q_{min}, \quad q[p-1] - \Delta q), \tag{11}$$

$$q_{max\_iter}[p] = min(q_{max}, \quad q[p-1] + \Delta q), \tag{12}$$

$$k_{j_{min\_iter}}[p] = max\left(k_{j_{min}}, \quad k_j[p-1] - \Delta k_j\right), \tag{13}$$

$$k_{j_{max\_iter}}[p] = min\left(k_{j_{max}}, \quad k_j[p-1] + \Delta k_j\right). \tag{14}$$



Sl. 4. Simulacioni rezultati za oblikovanje elipsoida krutosti na primeru krivolinijskog kretanja kada je zahtev da odnos osa elipsa bude minimalno 5:1: a) krutost u zglobovima; b) pozicija zglobova; c) norma (vrednost kriterijumske funkcije); d) greška orijentacije; e) CMS; f trajektorija i elipsoidi; g) trajektorija i elipsoidi- projekcija u xy ravni.

 ISBN 978-86-7466-930-3

To je računski zahtevan posao i nije ga jednostavno implementirati da radi u realnom vremenu, ali moguće ga je iskoristiti u fazi planiranja krutosti i trajektorije zglobova i završnog uređaja.

Simulacioni rezultati za primer krivolinijskog kretanja su prikazani na slici. Na slici 4a je prikazana krutosti u zglobovima, a na slici 4b pozicija zglobova. Slika 4c prikazuju vrednost (normu) optimizacione funkcije, dok slika 4d ilustruje grešku orijentacije. Slika 4e je CMS kojim su skalirane krutosti sa slike 4a da bi se dobili skalirani elipsoidi.

Primenjena optimizaciona funkcija i algoritam su ostvarili zadati cilj, da elipsoid ima željeno usmerenje i minimalni odnos osa. Dobijeni CMS skalira elipse da bi bile uporedive. Dobijene elipsoide moguće je dodatno skalirati da se dobije željena amplitude, ali samo po jednoj od osa. Da bi bilo moguće potpuno oblikovati elipsoid krutosti, potrebno je modifikovati kriterijumsku funkciju.

B. Oblikovanje elipsoida krutosti

Da bi se elipsoid krutosti potpuno oblikovala neophodno je da budu ostvareni sledeći zahtevi:

1) željeni pravac orijentacije elipsoida,
2) tačan odnos osa elipsoida,
3) magnitude osa elipsoida.

Kada su prva dva uslova zadovoljena, treći se ostvaruje primenom CMS-a na već oblikovanu elipsu. Da bi prva dva uslova mogla simultano da se optimizuju, predložena je modifikovana kriterijumska funkcija koja kombinuje grešku orijentaciju i odnos osa elipse

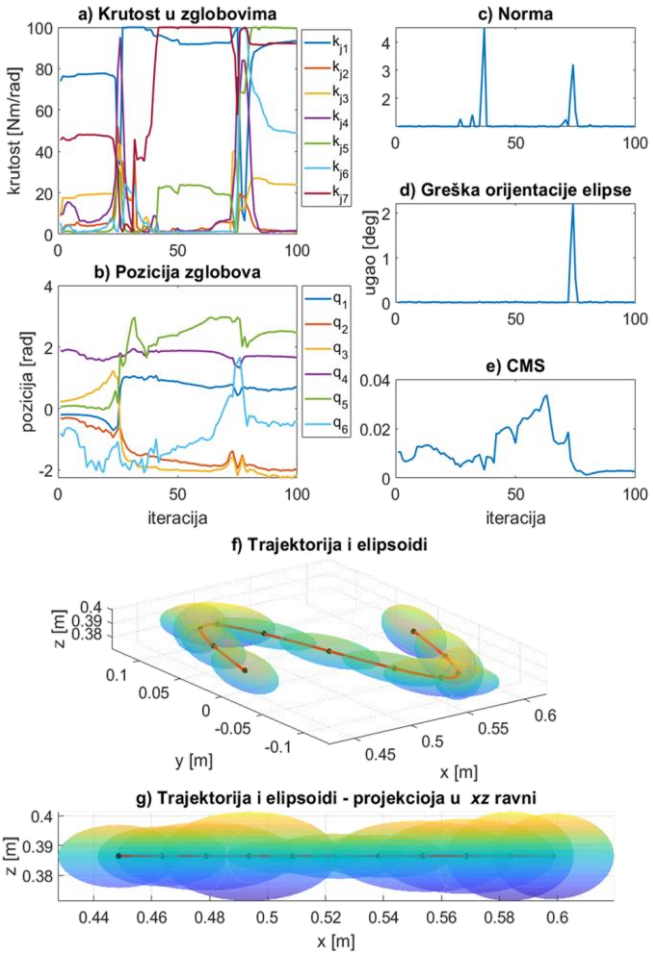
$$f_2 = (1 + |\alpha_{error}|)\left(1 + \left|r - \frac{r_{max}}{r_{min}}\right|\right). \quad (15)$$

Pozicije i krutosti zglobova u svakoj iteraciji se računaju identično kao i u prethodnom primeru opisanom u jednačinama (10) – (14) samo je optimizaciona funkcija promenjena i definisana kao f_2.

Funkcija je modifikovana tako da penalizuje svaku grešku orijentacije, i svako odstupanje odnosa osa od željenog odnosa. Zadati su zahtevi da je greška orijentacije $\alpha_{error} = 0°$, dok je željena orijentacija u smeru kretanja završnog uređaja robota, a odnos osa elipse $r = 5$.

Simulacioni rezultati za primer krivolinijskog kretanja su prikazani na slici 5. Na slici 5a je prikazana krutosti u zglobovima, a na slici 5b pozicija zglobova. Slika 5c prikazuju vrednost (normu) optimizacione funkcije, dok slika 5d ilustruje grešku orijentacije. Slika 5e je CMS kojim su skalirane krutosti sa slike 5a da bi se dobili skalirani elipsoidi.

Primenjena optimizaciona funkcija i algoritam su ostvarili zadati cilj, da elipsoid ima željeno usmerenje i odnos osa. Dobijeni CMS skalira elipse da bi bile uporedive. Dobijene elipsoide moguće je dodatno skalirati da se dobije željena amplitude. Algoritam je ostvario zadati cilj da elipsa ima željeno usmerenje i odnos osa. Dobijeni CMS skalira elipsoide da bi bile uporedivi. Rezultati pokazuju da svi elipsoidi imaju željeni oblik ili oblik koji je jako blizak



Sl. 5. Simulacioni rezultati za oblikovanje elipsoida krutosti na primeru krivolinijskog kretanja kada je zahtev da odnos osa elipsa bude tačno 5:1: a) krutost u zglobovima; b) pozicija zglobova; c) norma (vrednost kriterijumske funkcije); d) greška orijentacije; e) CMS; f trajektorija i elipsoidi; g) trajektorija i elipsoidi- projekcija u xy ravni.

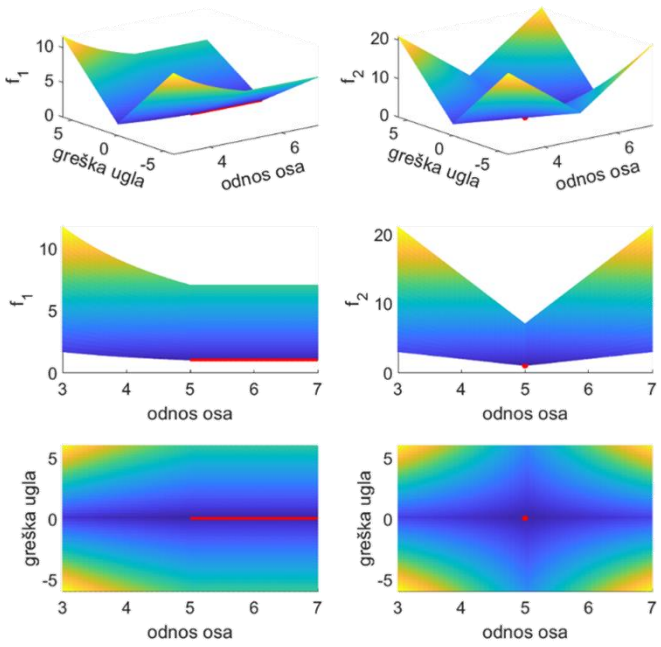
željenom. Dobijene elipsoide moguće je dodatno skalirati da se dobije željena amplitude osa.

Slika 6 ilustruje vrednosti funkcija u zavisnosti od greške orijentacije i odnosa osa elipse, a crvena boja označava minimum funkcije koji je u slučaju f_1 definisan kao prava, dok u slučaju f_2 kao jedna tačka. Samim tim je jasno da je optimalni set za f_2 koji je konzervativniji mnogo teže zadovoljiti nego u slučaju labavijih zahteva kriterijumske funkcije f_1. Ostvarivanje optimalnog rešenja u svakoj iteraciju uslovljeno je fizičkim ograničenjima samog robota.

V. DISKUSIJA I ZAKLJUČAK

U radu je uspešno implementiran pristup oblikovanja krutosti završnog uređaja robota pomoću grafičke reprezentacije krutosti preko elipsoida na modelu robota KUKA LWR. Krutost koja se analizira je zasnovana na modelu. Ilustrovano je da različiti kriterijumi daju rezultate različitog kvaliteta u skladu sa definisanim kriterijumima i fizičkim ograničenjima robota. Dobijeni rezultati uglavnom dobro ispunjavaju zadate zahteve, dok su ograničenja izraženija kod konzervativnije kriterijumske funkcije koja se ogleda u trenucima kada trajektorija ima najveće

 ISBN 978-86-7466-930-3



Sl. 6. Oblici kriterijumskih funkcija u zavisnosti od greške orijentacije i odnosa osa elipsi. Crvena boja označava minimum funkcije koji je u slučaju f_1 definisan kao prava, dok u slučaju f_2 kao jedna tačka

zakrivljenje, a samim tim i najveće promene orijentacije elipsoida. To pokazuje da su ostvarivi rezultati ograničeni izborom dela radnog prostora, kao i oblikom trajektorije. Klasična matrica translatorne krutosti ima 6 elemenata koje treba optimizovati, dok predložene kriterijumske funkcije koriste samo dva parametra, ugao greška ugla orijentacije elipsoida i odnos osa.

ZAHVALNICA

Rad je realizovan u okviru projekta Fonda za nauku Republike Srbije PROMIS, Grant #6062528, ForNextCobot i tokom istraživanja finansiranog od strane Ministarstva prosvete, nauke i tehnološkog razvoja Republike Srbije. Broj ugovora 2022/200103.

LITERATURA

[1] S. Haddadin, A. Albu-Schäffer and G. Hirzinger, "Safe Physical Human-Robot Interaction: Measurements, Analysis and New Insights," in *Robotics Research. Springer Tracts in Advanced Robotics*, Berlin, Heidelberg, Springer, 2010, pp. 395-407.

[2] A. De Santis, B. Siciliano, A. De Luca and A. Bicchi, "An atlas of physical human–robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253-270, 2008.

[3] N. Hogan, "Impedance Control: An Approach to Manipulation: Part II-implementation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, pp. 8-16, 1985.

[4] B. Lukić, T. Petrič, L. Žlajpah and K. Jovanović, "KUKA LWR Robot Cartesian Stiffness Control Based on Kinematic Redundancy," in *The 28th International Conference on Robotics in Alpe-Adria-Danube Region, RAAD 2019*, Kaiserslautern, Germany, 2019.

[5] R. Van Ham, T. G. Sugar, B. Vanderborght, K. W. Hollander and D. Lefeber, "Compliant actuator designs," *IEEE Robotics & Automation Magazine*, vol. 16, no. 3, pp. 81-94, 2009.

[6] A. Albu-Schäffer , M. Fischer, G. Schreiber, F. Schoeppe and G. Hirzinger, "Soft robotics: what Cartesian stiffness can obtain with passively compliant, uncoupled joints?," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004.

[7] F. Petit and A. Albu-Schäffer, "Cartesian impedance control for a variable stiffness robot arm," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, USA, 2011.

[8] M. H. Ang and G. B. Andeen, "Specifying and achieving passive compliance based on manipulator structure," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 4, pp. 504-515, 1995.

[9] N. Lukić and P. B. Petrović, "Complementary projector for null-space stiffness control of redundant assembly robot arm," *Assembly Automation*, vol. 39, no. 4, pp. 696-714, 2019.

[10] N. Knežević, B. Lukić, K. Jovanović, L. Žlajpah and T. Petrič, "End-effector Cartesian stiffness shaping - sequential least squares programming approach," *Serbian Journal of Electrical Engineering*, vol. 18, no. 1, pp. 1-14, 2021.

[11] N. Knežević, B. Lukić and K. Jovanović, "Feedforward Control Approaches to Bidirectional Antagonistic Actuators Based on Learning," in *Advances in Service and Industrial Robotics*, Cham, Springer, 2020, pp. 337-345.

[12] W. Roozing, Z. Ren and N. G. Tsagarakis, "An efficient leg with series–parallel and biarticular compliant actuation: design optimization, modeling, and control of the eLeg," *The International Journal of Robotics Research*, vol. 40, no. 1, pp. 37-54, 2021.

[13] A. Nejadfard, S. Schütz, K. Mianowski, P. Vonwirth and K. Berns, "Moment Arm Analysis of the Biarticular Actuators in Compliant Robotic Leg Carl," in *Biomimetic and Biohybrid Systems*, Cham, Springer, 2018, pp. 348-360.

[14] A. Ajoudani, N. G. Tsagarakis and A. Bicchi, "On the role of robot configuration in Cartesian stiffness control," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, 2015.

[15] A. Ajoudani, N. G. Tsagarakis and A. Bicchi, "Choosing Poses for Force and Stiffness Control," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1483-1490, 2017.

[16] A. Ajoudani, M. Gabiccini, N. Tsagarakis, A. Albu-Schäffer and A. Bicchi, "TeleImpedance: Exploring the role of common-mode and configuration-dependant stiffness," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, Osaka, Japan, 2012.

[17] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming for large-scale nonlinear optimization," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 123-137, 2000.

[18] N. Knežević, B. Lukić, K. Jovanović , T. Petrič and L. Žlajpah, "End-Effector Cartesian Stiffness Optimization: Sequential Quadratic Programming Approach," in *Proceedings of the 5th International Conference on Electrical, Electronic and Computing Engineering (IcETRAN 2019)*, Silver Lake, Serbia, 2019.

ABSTRACT

The stiffness of the robot's end-effector (EE) determines the robot's behavior when interacting with the environment. The stiffness of the EE is most often represented by a stiffness matrix whose complete design can be impossible because robots do not have enough degrees of freedom to adjust all the elements in the stiffness matrix. This leads to the application of optimization that balances the values of the stiffness matrix elements that are of interest for the task. In this paper, an approach for offline shaping of the stiffness matrix using stiffness ellipsoids is proposed, where stiffness is shaped by adjusting the orientation and axis length of the ellipsoid. The shape of the ellipsoid has fewer parameters that need to be adjusted relative to the stiffness matrix. Two criterion functions for shaping the ellipsoid stiffness have been proposed to exploit robot kinematic redundancy in shaping the stiffness ellipsoid. Optimal values of joint positions and stiffness were calculated using the algorithm based on SLSQP (Sequential Least SQuare Programming). Following the shaping of EE stiffness ellipsoid, adjustments or its volume based on variations in joint stiffness are highlighted.

End-effector passive stiffness shaping through the stiffness ellipsoid

Branko Lukić, Nikola Knežević i Kosta Jovanović

 ISBN 978-86-7466-930-3

Hijerarhijsko distribuirano upravljanje kolaborativnim industrijskim humanoidnim robotom podržano oblak-arhitekturom

Jovan Šumarac, Ilija Stevanović, Aleksandar Rodić

***Apstrakt*—Ovaj rad predstavlja jednu hijerarhijsku, distribuiranu upravljačku arhitekturu dvoručnog, kolaborativnog, humanoidnog servisnog robota podržanom tehnologijom "računarstva u oblaku". Ova upravljačka struktura dizajnirana je namenski za izvršavanje robotskih zadataka koji zahtevaju primenu složenijih manipulativnih i kognitivnih veština u okviru industrijskog proizvodnog tehnološkog procesa. Upravljačka struktura robota osmišljena je tako da zadovolji potrebe tzv. pametne proizvodnje u okviru platforme Industrije 4.0, odnosno da omogući i olakša kooperaciju čoveka i robota kao njemu komplementarnog inteligentnog sistema, neophodnog u industrijskim zadacima koje čovek ne može samostalno sprovoditi. Ovaj rad prikazuje strukturu i način funkcionisanja sistema hijerarhijskog distribuiranog upravljanja, organizovan na tri operativna nivoa: a) strateškom, b) taktičkom i c) izvršnom. To je omogućeno zahvaljujući korišćenju arhitekture oblak-računarskog sistema. Hardver upravljačkog sistema uspostavljen je na tri nivoa: fizičkom, komunikacionom i aplikativnom. Zahvaljujući distribuiranoj arhitekturi, sistem oblaka omogućava distribuirano izvršavanje pojedinačnih upravljačkih zadataka što rezultira uravnoteženim opterećenjem multiprocesorskog sistema i smanjenjem vremena odziva na stimuluse iz fizičkog okruženja robota.**

***Ključne reči*—upravljanje u oblaku, kolaborativni roboti, distribuirana inteligencija, pametna proizvodnja, Industrija 4.0**

I. UVOD

Od kada je kompanija „General Motors" u Detroitu (SAD) 1961. napravila prvog industrijskog robota za manipulaciju (UNIMATE) i koristila ga u automobilskoj industriji za sklapanje delova, timovi naučnika, inženjera i tehnologa razmišljali su o tome kako da automatizuju ljudske manipulacione i kognitivne veštine što je više moguće i kako da robote učine bezbednim i kolaborativnim. Prethodne generacije industrijskih robota za manipulaciju, kao deo svojih kontrolera, koristile su algoritme za "imitaciju" ljudskih manipulativnih veština uz korišćenje petlji povratne sprege po poziciji ondosno brzini i po sili odnosno momentu. Ovo je uglavnom bilo dovoljno za obavljanje tehnoloških operacija u okviru proizvodnih procesa u visoko tehnološki strukturiranim infrastrukturnim okruženjima – tehnološkim proizvodnim linijama. Međutim, takav pristup je imao relativno nizak nivo fleksibilnosti i prilagodljivosti poremećajima i promenama u proizvodnim procesima.

Sa napretkom informacionih tehnologija, bežičnih komunikacija, "Interneta stvari", senzorskih tehnologija i mašinske vizije, otvorene su nove perspektive automatizacije tehnoloških procesa tako što je industrijskim robotima omogućeno da, sa svojstvenom preciznošću i brzinom, ostvare sofisticirane atribute ljudske mikro-manipulativne i taktilne veštine na visokom nivou kognitivne percepcije i sposobnosti rasuđivanja [1].

Cilj ovog rada jeste da predstavi kako se ljudske veštine i znanje kvalifikovanih radnika mogu sticati, čuvati i koristiti u bazi podataka na sistemu oblaka (ondosno "Cloud" sistemu), odakle se ovi podaci mogu analizirati i koristiti za obuku (tj. za učenje veština) industrijskog humanoidnog robota u izvođenju različitih manipulativnih i kolaborativnih zadataka u saradnji sa ljudima. U tu svrhu razvijen je originalni dvoručni kolaborativni robotski sistem (industrijski humanoidni servisni robot) koji omogućava implementaciju i verifikaciju različitih algoritama za automatizaciju ljudskih manipulativnih i kognitivnih veština u proizvodnim procesima [2]. Originalni robotski sistem sa svim hardverskim i softverskim komponentama razvijen je u Centru za robotiku Instituta "Mihajlo Pupin".

II. INDUSTRIJSKI HUMANOID

A. Mehanički dizajn prototipa robota

Industrijski humanoid je dvoručni (ili bi-manuelni) kolaborativni servisni robot namenjen industrijskim zadacima [3]-[7], koji po svojoj strukturi podseća na svoj biološki model – čoveka. U mehaničkom smislu industrijski humanoid ima sledeće module: (i) sistem podrške za sletanje montiran na mobilnu platformu (motorizovana kolica), (ii) segmentirani trup - torzo robota, (iii) dve industrijske robotske ruke sa odgovarajućim hvatačima (Sl. 1).

Humanoidni robot je montiran na motorizovana kolica sa diferencijalnim upravljanjem na par pogonskih (aktivnih) točkova. Kolica imaju i dva pomoćna (pasivna) točka, koja se okreću oko sopstvenih vertikalnih osa. Zahvaljujući takvoj strukturi, kolica robota imaju mogućnost da se kreću u tri koordinatna pravca: napred-nazad, levo-desno i da se okreću

Jovan Šumarac – Centar za Robotiku, Institut "Mihajlo Pupin", Univerzitet u Beogradu, Volgina 15, 11060 Beograd, Srbija (e-mail: jovan.sumarac@pupin.rs).

Aleksandar Rodić – Centar za Robotiku, Institut "Mihajlo Pupin", Univerzitet u Beogradu, Volgina 15, 11060 Beograd, Srbija (e-mail: aleksandar.rodic@pupin.rs).

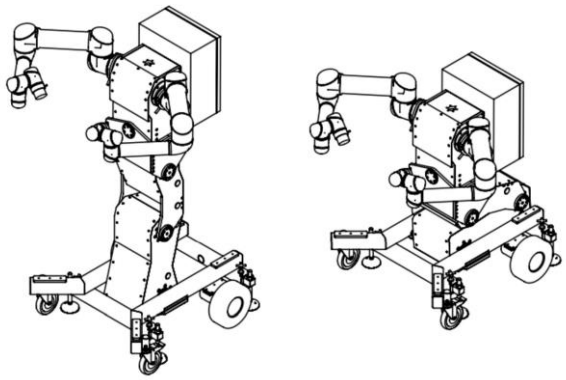
Ilija Stevanović – Centar za Robotiku, Institut "Mihajlo Pupin", Univerzitet u Beogradu, Volgina 15, 11060 Beograd, Srbija (e-mail: ilija.stevanovic@pupin.rs).

 ISBN 978-86-7466-930-3

oko sopstvene ose u opsegu od 0 do 360 stepeni. Nosivi sistem bimanuelnog robota je tehnička varijanta (imitacija) ljudskih nogu. Mehanizam za noge robota može se savijati oko tri ose. Jedna osa rotacije prolazi kroz skočni zglob, druga osa kroz zglob kolena, a treća kroz zglob kuka. Na ovaj način se sinhronizovanim pokretima u pomenutim zglobovima može podesiti položaj (orijentacija) trupa humanoida u odnosu na površinu oslonca.

Gornji deo trupa robota (torzo) ima mogućnost savijanja oko sagitalnog pravca u frontalnoj ravni (levo-desno). Uz mogućnost promene nagiba trupa napred-nazad u sagitalnoj ravni (promenom konfiguracije nogu), ovo omogućava humanoidnom robotu da se prilagodi svom operativnom prostoru zadatka.

Kolaborativni dvoručni robot ima dve industrijske robotske ruke UR5 sa odgovarajućim hvatačima koje mu daju mogućnost različitih manipulacija objektima kao i fizičke interakcije sa okolinom. Takođe, svaka robotska ruka ima instaliran zglobni senzor sile/momenta na završnom uređaju.



Sl. 1. 3D prototip modela industrijskog robota.

Visina ovog humanoida iznosi oko 175cm a njegov raspon ruku iznosi oko 2300 mm. Ideja iza projektovanja ovakvog sistema jeste da njegove dimenzije budu približne čovekovim kako bi robot mogao lako da zameni čoveka u određenim zadacima, ali i da bi mogao prirodno da obavlja zadatke u kojima postoji kolaboracija sa čovekom. Analiza njegovog radnog prostora pokazala je dobru manipulativnost ovog sistema u oblastima zajedničkog radnog prostora obe robotske ruke, pogotovo u zonama ispred samog torzoa robota [8].

Na Sl. 1. može se videti i ormarić koji je okačen na robotski torzo kao svojevrsni "ranac". U njemu se nalaze različite elektronske komponente neophodne za funkcionisanje celokupnog sistema (mikrokontroleri, uključujući i glavni mikrokontroler ondnosno "mozak" sistema, zatim napajanja, bezbednosne sklopke, releji i dr.).

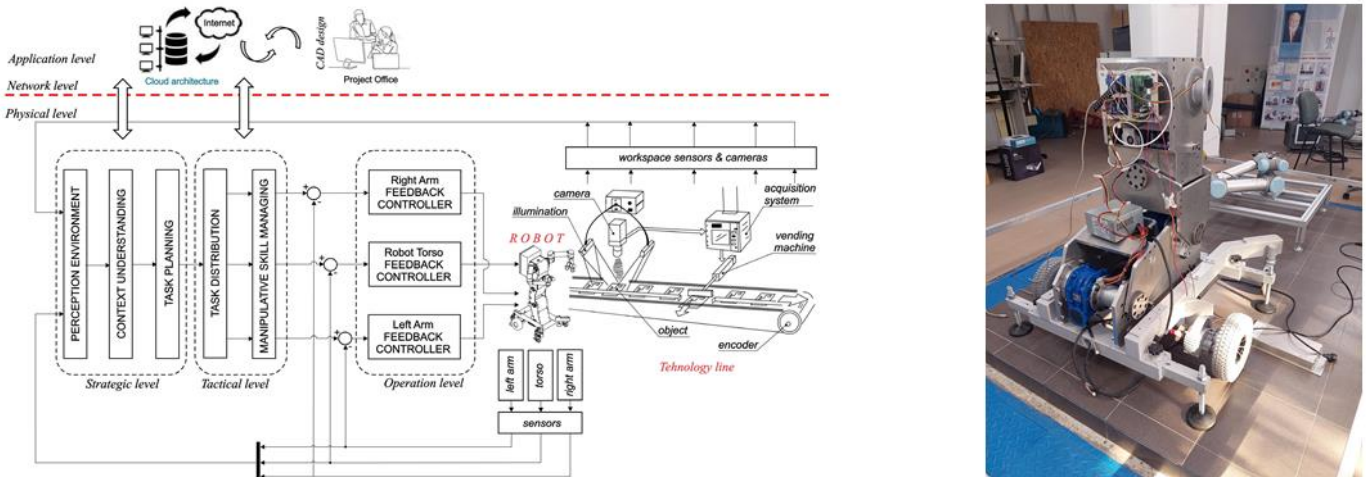
B. Arhitektura hardvera kontrolera robota

Hardverska struktura celokupnog sistema upravljanja industrijskog servisnog robota prikazanog na Sl. 1 je projektovana kao distribuirana računarsko-senzorska mreža [9] koja funkcioniše na tri funkcionalna nivoa (Sl. 2): (i) fizički nivo, (ii) nivo mreže i (iii) aplikativni nivo.

Na fizičkom nivou, postoji robot sa pripadajućim senzorima i kamerama, poseban kontroler trupa robota, kao i nezavisni kontroleri robotskih ruku i pridruženih robotskih hvatača. Globalno posmatrano, to je prilično složena struktura dizajnirana da omogući specifične sposobnosti robota. Takođe na fizičkom nivou, u okruženju robota postoje odgovarajuće mašine alatke sa svojim upravljačkim jedinicama i na kraju ljudi (radnici) koji zajedno sa dvoručnim robotom čine deo tehnološkog procesa.

Mrežni, odnosno komunikacioni nivo hardverske strukture "Cloud" kontrole, zadužen je za brzu i efikasnu međusobnu komunikaciju između sastavnih komponenti robotske ćelije i odgovarajućeg tehnološkog procesa. Uloga mreže je da obezbedi pouzdan prenos informacija i senzorskih podataka do svih i od svih konstitutivnih elemenata sistema. Komunikacioni nivo dakle obezbeđuje umrežavanje robota u lokalnu računarsku mrežu (LAN) korišćenjem Wi-Fi pristupa ili GSM-GPRS modema (u slučaju informaciono strukturiranog okruženja) i povezivanje robota na Internet.

Na nivou aplikacije postoji nekoliko računara koji međusobno komuniciraju i koriste zajedničku bazu podataka. Svaki, pojedinačni računar ima strogo definisane zadatke. Jedan obrađuje podatke koji se odnose na karakteristike objekata (delova) i njima manipuliše, obrađuje podatke koji se odnose na mašine alatke preuzete iz okoline, njihove dimenzije, prostornu lokaciju itd. Drugi računar je zadužen za



Sl. 2. Blok dijagram hardverske strukture robotskog kontrolera industrijskog humanoida (levo). Sastavljeni prototip robota (desno)..

 ISBN 978-86-7466-930-3

analizu, učenje i operacionalizaciju ljudskih manipulativnih i kognitivnih veština koje su karakteristične za tehnološke procese. Treći računar se bavi tehnikama i metodama formalizacije određenih industrijskih tehnoloških procesa (npr. obrada, montaža delova, fizička pomoć ljudskim radnicima, itd.) na način koji će biti razumljiv za mašinu (kontroler robota). Informacije koje se odnose na tehnološke operacije se uglavnom svode na podatke kojim redosledom i kojim alatima treba izvršiti određene tehnološke operacije, na koji način, u kojoj tački uhvatiti objekat, kojom silom ili momentom, kojom brzinom to učiniti, itd.

Zadaci koji se obavljaju na računarima, koji su deo aplikativnog nivoa, u velikoj meri se oslanjaju na baze podataka u vezi sa aplikacijama koje su operativni domen industrijskog humanoida. Baze podataka sadrže informacije koje se odnose na geometrijske atribute objekata koji su predmet tehnološkog procesa, zatim informacije od interesa o tehnikama ljudskih veština karakterističnim za iskusne radnike (stručnjake) i informacije o fazama ili redosledu izvođenja pojedinih tehnoloških operacija i detalje o tome kako se one izvode sa specifičnim numeričkim indikatorima (kompozitna brzina, pozicija, sila, obrtni moment, itd.).

Ljudskim operaterima "Cloud" arhitektura omogućava daljinsko upravljanje robotom pomoću pametnog telefona sa Android aplikacijom ili standardnog PC računara sa odgovarajućom aplikacijom sa grafičkim korisničkim interfejsom. Ovo pretpostavlja Wi-Fi hot spot ili GSM-GPRS modem na robotu koji donosi dodatnu funkcionalnost daljinske komunikacije sa ovlašćenim pristupom i mogućnošću praćenja parametara sistema i/ili daljinskog prenosa slike i zvuka.

III. UPRAVLJAČKI NIVOI

Za upravljanje dvoručnim kolaborativnim robotom prikazanim na Sl. 1, koristi se arhitektura upravljanja u oblaku (odnosno "Cloud" kontrola) prikazana na Sl. 2. Ova kontrolna konfiguracija podržava koncept takozvane hijerarhijski distribuirane kontrole implementirane na tri nivoa kontrole: (i) strateški, (ii) taktički i (iii) izvršni nivo. Svi navedeni nivoi upravljanja imaju precizno diferencirane kontrolne zadatke i postoji jasna podređenost operacija u sistemu.

A. Strateški nivo

Na strateškom nivou upravljanja definišu se globalni ciljevi, planiranje procesa, tehnološke operacije koje robot treba da obavlja samostalno ili u saradnji sa čovekom ili mašinom, kao i manipulativne i kognitivne veštine koje robot treba da demonstrira dok obavlja svoje aktivnosti, poštujući redosled tehnoloških operacija i bezbednosne standarde. Strateška kontrola koristi informacije dobijene na fizičkom nivou (Sl. 2) kao i iz odgovarajuće baze podataka. U bazama podataka se čuvaju informacije relevantne za robota industrijske usluge koje se odnose na prepoznavanje geometrije objekata, informacije o fazama i tehnikama izvođenja određenih tehnoloških procesa u koje je robot uključen i na kraju informacije o manipulativnim i kognitivnim veštinama pretvorene u formu razumljivu mašini (robotski kontroler). U bazi se čuvaju i podaci o bezbednosnim standardima za obavljanje određenih robotskih zadataka, budući da je industrijski humanoid namenjen za kolaborativni rad sa ljudima. Takođe, na nivou strateške kontrole vodi se računa o ispunjavanju standarda kvaliteta izvođenja tehnoloških operacija. Rezultati ovog nivoa kontrole su jasno definisani ciljevi visokog nivoa i redosled ostvarivanja tehnološkog procesa, tehničko-tehnološka ograničenja, način izvođenja procesa, alati koji se koriste i odgovarajuće veštine, razvrstane na podciljeve koje treba postići na nižem, podređenom hijerarhijskom nivou – taktičkom.

B. Taktički nivo

Na nivou taktičke kontrole, kako sama reč kaže, definiše se taktika ostvarivanja skupa podciljeva dobijenih sa strateškog nivoa upravljanja. Na ovom nivou planira se postizanje skupa podciljeva kao što su planiranje putanje robotskih ruku (pozicija, orijentacija i brzina krajnjih efektora robota), sinhronizacija rada dve robotske ruke kako bi se izbegao sudar, definisanje referentne sile i momenta na hvataču robota, zatim, orijentisanje kamera u radnom prostoru zadatka, itd. Izlazni podaci sa nivoa taktičke kontrole predstavljaju referentne (tj. željene) vrednosti promenljivih koje predstavljaju ulaze za izvršni nivo upravljanja (na servo nivou).

C. Izvršni nivo

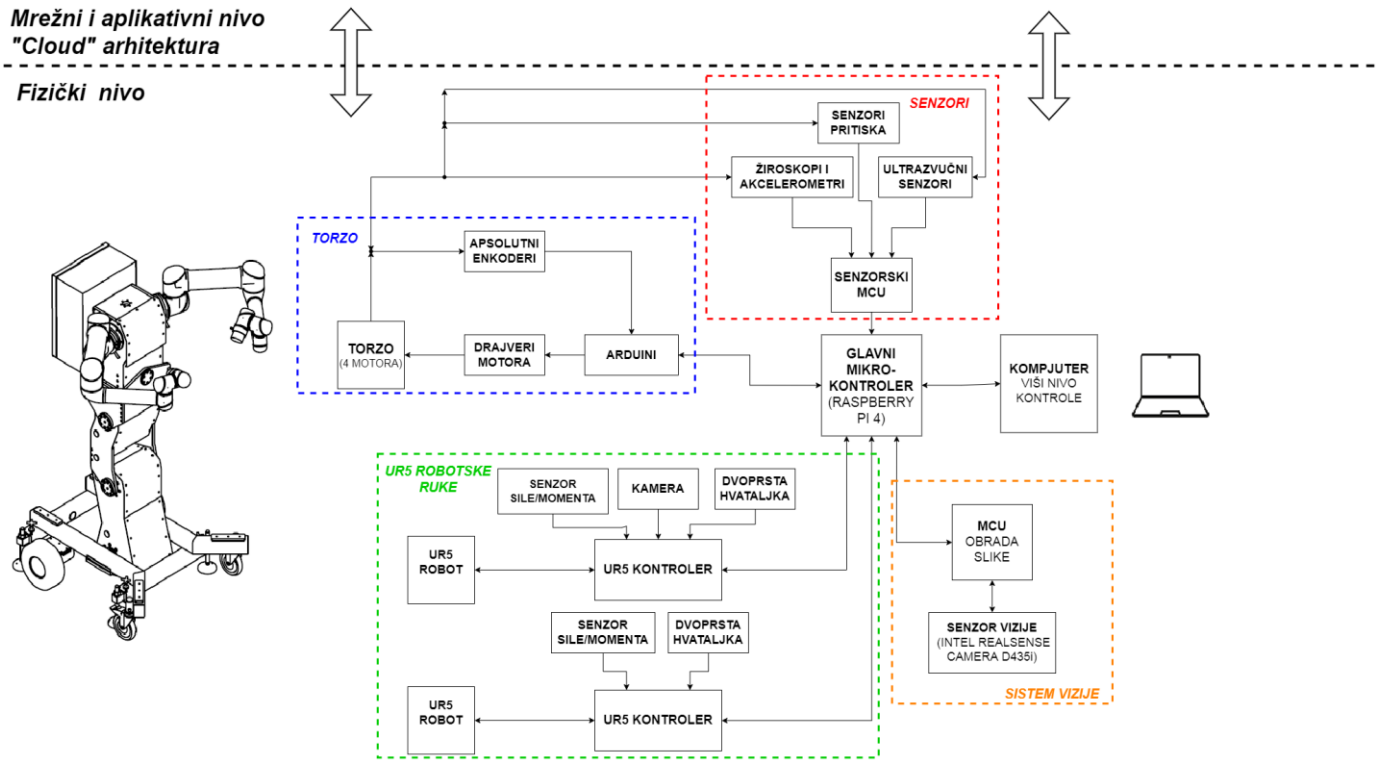
Na izvršnom nivou upravljanja komanduje se radom servo pogona koji obezbeđuju izvršavanje određenih aktivnosti. Korišćenjem povratnih petlji po poziciji/orijentaciji, brzini, sili/momentu, na izvršnom nivou postiže se željeno kretanje sistema u skladu sa taktikom i strategijom izvršavanja procesa koji su određeni na višim nivoima sistemske kontrole. Na izvršnom nivou, senzorne informacije i podaci senzora prikupljaju se sa robota i/ili iz njegovog fizičkog okruženja. Takođe se obavlja akvizicija slike i/ili video tokova iz sistema vizije robota za njihovo slanje na viši, taktički nivo upravljanja gde se vrši obrada tih signala. Dobijene povratne informacije su od velikog značaja na taktičkom nivou za vršenje neophodnih korekcija tehnike izvođenja procesa redefinisanjem referentnih vrednosti. Na ovaj način se obezbeđuje tok informacija odozgo nadole, kao i inverzni tok informacija odozdo prema gore od najvišeg nivoa upravljanja ka najnižem i obrnuto.

D. Upravljanje robotom na fizičkom nivou "Cloud" sistema

Na Sl. 3 prikazan je blok dijagram upravljanja dvoručnim kolaborativnim robotom. Upravljanje izvršnim nivoom se realizuje korišćenjem dva nezavisna industrijska kontrolera robotskih ruku UR5 koji se isporučuju sa svojim fabričkim kontrolerima zatvorenog tipa sa već realizovanim algoritamskim rešenjima za regulator položaja/brzine i regulator sile/momenta.

Torzo robota, robotske ruke i sistem vizije se nezavisno kontrolišu, dok se centralna mikrokontrolerska jedinica (MCU) koristi za sinhronizaciju celog upravljačkog sistema,

 ISBN 978-86-7466-930-3



Sl. 3. Blok šema upravljačkog sistema razvijena na fizičkom nivou industrijskog servisnog robota upravljanog "iz oblaka".

delegiranje zadataka, komunikaciju sa udaljenim operaterima, itd. Torzo robota (Sl. 2) ima četiri zgloba aktuirana DC motorima bez četkica koji se pokreću preko drajvera. Svaki od drajvera kontroliše nezavisni Arduino mikrokontroler. Na svaki motor montiran je rotacioni enkoder koji pruža informacije o trenutnoj ugaonoj brzini i položaju motora, koje Arduino koristi za kontrolu položaja motora. Arduino mikrokontroleri komuniciraju sa glavnim MCU putem CAN bus komunikacije velike brzine. Na torzo i kolica robota postavljeni su različiti senzori uključujući i senzore kontakta/pritiska, žiroskope i akcelerometre, ultrazvučne senzore, itd. Senzori pritiska su posebno važni jer se koriste za izračunavanje tačke nultog momenta (ZMP) mobilnog industrijskog humanoida kao indikatora stabilnosti. Ove informacije se koriste da bi se utvrdilo da li je bezbedno da torzo i ruke robota stignu u datu poziciju bez pada ili prevrtanja. Sirove informacije sa senzora se obrađuju na posebnoj mikroprocesorskoj jedinici, a potom se obrađeni podaci, po potrebi, CAN komunikacijom šalju glavnom MCU, koji ih koristi za kontrolne zadatke, mere bezbednosti, itd. Ruke robota UR5 su fizički montirane na prirubnicama koje se nalaze na vrhu torza. Prirubnice su zakrivljene pod uglom od oko 15 stepeni u odnosu na torzo robota jer se time povećava opseg korisnog radnog prostora ispred torzoa.

UR5 roboti su konvencionalne industrijske ruke sa 6 stepeni slobode. Svaka od robotskih ruku ima senzor sile/momenta i hvatač sa dva prsta koji je montiran na svom vrhu. Jedna od ruku takođe ima montiranu kameru na zglobu i ovo će biti glavna ruka u dvoručnim zadacima. Servo kontrola niskog nivoa zglobova ruku robota se vrši preko originalnih UR5 kontrolera koji se isporučuju zajedno sa robotskim rukama. UR5 kontroleri komuniciraju sa MCU preko fizičkih veza (I/O signali sa MODBUS vezom), preko žičane (Ethernet) ili bežične (Wi-Fi pristup) mreže i preko ROS-a (Robot Operating Sistem).

Sistem robotske vizije je važan kontrolni podsistem koji celokupnom sistemu obezbeđuje vizuelnu povratnu informaciju. Senzor vida (Intel RealSense D435i dubinska kamera) je montiran na vrhu torzoa. Njegova uloga je da obezbedi robotu mogućnost 3D vizije sa senzorom dubine. Takav sistem se koristi za opažanje robotskog okruženja, praćenje zadataka, izbegavanje fizičkih stacionarnih ili mobilnih prepreka itd. Kamera će biti povezana sa posebnom mikrokontrolerskom jedinicom koja će se koristiti za obradu slika primljenih sa kamere. Softver koji će se koristiti za procesiranje vizuelnih podataka sastoji se kako iz klasičnih algoritama kompjuterske vizije tako i iz algoritama dubokog učenja i neuralnih mreža za obradu i prepoznavanje slika. Ovaj mikrokontroler će takođe biti povezan na glavni MCU.

Glavna jedinica mikrokontrolera (Raspberry Pi 4 sa 8GB RAM-a) je sastavni deo kontrolnog sistema. Ovaj MCU komunicira sa mrežnim nivoom (slika 2), odakle prima informacije o referentnim zadacima robotskog sistema koje treba da izvrši. Takođe prima povratne informacije od svakog kontrolnog podsistema. MCU obrađuje sve ove informacije i sinhronizuje ceo sistem (torzo+2 ruke+2 hvatača+kamera), delegira zadatke svakom podsistemu i zaustavlja aktuatore u slučaju opasnosti ili kvara. Komplementarni računar se može dodati glavnom MCU kao lokalni nadzorni sistem. Koristi se uglavnom kao pomoćno sredstvo za razvoj sistema

 ISBN 978-86-7466-930-3

upravljanja, testiranje i verifikaciju rezultata rada zadataka.

IV. ZAKLJUČAK

U radu je predložena višestepena upravljačka arhitektura servisnog industrijskog kolaborativnog robota. Hijerarhijska distribuiranost ovog sistema dozvoljava najbolje iskorišćenje pojedinačnih kontrolnih podsistema kao i već postojećih kontrolera i stavlja fokus na delegiranje zadataka na manje procesorske jedinice i sinhronizaciju pojedinačnih delova sistema. Korak dalje u dizajniranju ovakvih sistema predstavlja implementacija "Cloud" arhitekture koja omogućava daljinsko upravljanje i nadzor nad sistemom, kao i čuvanje raznih ulaznih i izlaznih sistemskih podataka na serveru.

Dalji rad autora nastaviće se u ovom pravcu. Po finalnoj montaži celokupnog sistema radiće se eksperimentalna verifikacija različitih upravljačkih zadataka. Potom će gotov sistem biti pogodan za različita istraživanja u oblasti dvoručne manipulacije, kolaboracije čoveka i robota, inteligentnog hvatanja, robotske vizije i dr.

ZAHVALNICA

Razvoj prototipa kolaborativnog robota rezultat je bilateralnog R&D projekta pod nazivom „Razvoj i eksperimentalna verifikacija performansi mobilnog robota sa dve ruke za saradnju sa čovekom" iz „Programa razvoja nauke i tehnologije – finansiranje zajedničkih istraživačko-razvojnih projekata". R. Srbija i NR Kina", 2017-2019.

LITERATURA

[1] A. Rodić, I. Stevanović, M. Jovanović, „Smart Cyber-Physical System to Enhance Flexibility of Production and Improve Collaborative Robot Capabilities – Mechanical Design and Control Concept", Proceedings of the 27th International Conference on Robotics in Alpe-Adria Danube Region (RAAD 2018), In book: Advances in Service and Industrial Robotics, Eds. Nikos A. Aspragathos, Panagiotis N. Koustoumpardis, Vassilis C. Moulianitis, Part of the Mechanisms and Machine Science book series (Mechan. Machine Science, Vol. 67), pp. 627-639, 2018

[2] A. Rodić, J. Šumarac, I. Stevanović, M. Jovanović, "Cloud-Enabled Bi-manual Collaborative Robot with Enhanced Versatility for Customized Production", Proceedings of the 30th International Conference on Robotics in Alpe-Adria-Danube Region, Futuroscope-Poitiers, France, pp. 240-249, June, 2021.

[3] N. Kashiri et al., "CENTAURO: A Hybrid Locomotion and High Power Resilient Manipulation Platform," in IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 1595-1602, April 2019, doi: 10.1109/LRA.2019.2896758.

[4] M. Fuchs et al., "Rollin' Justin - Design considerations and realization of a mobile platform for a humanoid upper body," 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 4131-4137, doi: 10.1109/ROBOT.2009.5152464..

[5] S. S. Srinivasa et al., "Herb 2.0: Lessons Learned From Developing a Mobile Manipulator for the Home," in Proceedings of the IEEE, vol. 100, no. 8, pp. 2410-2428, Aug. 2012, doi: 10.1109/JPROC.2012.2200561.

[6] A. Rodić, B. Miloradović, S. Popić, S. Spasojević, B. Karan, "Development of Modular Compliant Anthropomorphic Robot Hand", In Book: New Trends in Medical and Service Robots. Theory and Integrated Ap- plications, Series: Mechanisms and Machine Science, Springer Publish- ing House, Vol. 16, Pisla, D.; Bleuler, H.; Rodic, A.; Vaida, C.; Pisla, A. (Eds.), 2014, VIII, 238 p. 167, ISBN 978-3-319-01591-0, Due: Sep- tember 30, (2013).

[7] A. Rodić, B. Miloradović, S. Popić, Đ. Urukalo, "On developing lightweight robot-arm of anthropomorphic characteristics". In Book: New Trends in Medical and Service Robots. Book 3, Series: Mechanisms and Machine Science, Springer Publishing House, Vol. 38, Bleuler, H.; Pisla, D.; Rodic, A.; Bouri, M; Mondada, F; (Eds.), ISBN: 978-3-319- 23831-9, Book ID: 332595 _1_En, (2015).

[8] J. Šumarac, K. Jovanović, A. Rodić, "Workspace Analysis of a Collaborative Bi-manual Industrial Robotic System" ROI1.1, In Proceedings of the IcETRAN2021, Banja Vrućica, Bosnia and Herzegovina, September, (2021)

[9] A. Rodić, M. Jovanović, S. Popić, G. Mester, "Scalable Experimental Platform for Research, Development and Testing of Networked Robotic Systems in Information Structured Environments". In Proceedings of the IEEE SSCI2011, Symposium Series on Computional Intelligence, Workshop on Robotic Intelligence in Information Structured Space, pp. 136-143, Paris, France, (2011).

ABSTRACT

This paper presents a multi-level cloud-enabled control architecture of a bi-manual collaborative humanoid service robot. This architecture is designed to perform robotic tasks that require the implementation of advanced manipulative and cognitive skills within the industrial manufacturing process. The control structure of the robot is designed to meet the needs of the so-called smart production within Industry 4.0, enough to enable and facilitate the cooperation of man and robot as a complementary intelligent system, necessary in industrial tasks that humans cannot perform independently. This paper presents a hierarchically distributed management system organized into three operational levels: a) strategic, b) tactical and c) executive, enabled by the use of cloud architecture. The hardware of the control system is established on three levels: physical, communication and application. Thanks to the distributed architecture, the cloud system enables distributed execution of individual control tasks, which results in a balanced load on the multiprocessor system and reduced stimulus response time from the physical environment of the robot.

Hierarchically Distributed Control of Collaborative Industrial Humanoid Robot Supported by Cloud-Computing Architecture

Jovan Šumarac, Aleksandar Rodić, Ilija Stevanović