Development of applicative interface for connecting optical 3D scanner and robot controller of the UR-5 industrial robot arm

Emilija Stanković, School of Electrical Engineering, University of Belgrade
Ilija Stevanović, Institute Mihajlo Pupin, Robotics department
Aleksandar Rodić, Institute Mihajlo Pupin, Robotics department

***Abstract*— This paper is focused on the implementation, development and testing of an interface for connecting an automated 3D scanner and an industrial robot. Scanning is based on the use of Structured Light scanning software and the accompanying equipment. A rotating platform is constructed, and it is powered by an Arduino-controlled stepper motor. Program in addition to rotating platform by optimal number of degrees, also communicates with the scanning software - via the COM port to synchronize the movement of the rotating platform and the scan. 3D image is obtained by connecting several captured frames during the rotation of the object. The result is point cloud in space, post-processing is performed, and selected points form a robot trajectory. Simulation is performed by MATLAB Robotics Toolbox.**

***Index Terms*— Structured Light scanning, Robotics System Toolbox, UR-5, Point cloud**

I. INTRODUCTION

Spatial visualization through realistic 3D modeling is the process of obtaining a mathematical representation of the three-dimensional surface of an object. 3D scanning is widely used in all spheres of industry because it provides fast data collection, and it guarantees a high quality. 3D scanners have the ability to capture in detail points, lines and surfaces of objects, as well as texture and size.

3D scanning is transforming nearly every industry, construction, manufacturing, marketing, medical industry, forensics, reverse engineering in areas of automotive, aerospace and shipbuilding. It has a multitude of applications in robotic vision and guidance, some of them being AI applications, art, design, interior visualization, media, machine control, site layout, flight and wind tunnel testing, prototyping and 3D printing, quality control, and scanning of power equipment assembling of a substation.[1]

Emilija Stanković is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: emilijazstankovic@gmail.com)

Ilija Stevanović is with the Institute Mihajlo Pupin, University of Belgrade, 15 Volgina, 11020 Belgrade, Serbia (e-mail: ilija.stevanovic@pupin.rs)

Aleksandar Rodić is with the Institute Mihajlo Pupin, University of Belgrade, 15 Volgina, 11020 Belgrade, Serbia (e-mail: aleksandar.rodic@pupin.rs)

With that in mind, incorporating 3D scanning technology and robotic manipulators has the potential to facilitate all spheres of industry, from simple things like manufacturing to complicated operations like robotic surgery. Different approaches to using 3D scanners as robotic vision systems have been proposed in the literature.[2][3][4] In industry, robotic systems are taught by guidance to perform specific tasks on object. When changing the object that the manipulator needs to interact with it is necessary to repeat the process of teaching guidance every time. With the introduction of 3D scanning this problem has been avoided and flexibility precision and repeatability are increased. Nowadays, robotic surgery occupies an important place when it comes to the advancement of technology and is being applied more and more every day. Some of the fields are Microsurgery (Micro-manipulation), Skin harvesting (Surface tracking), Neurosurgery Interventional radiology (Constrained targeting). [5]

II. DEVELOPMENT GOALS

Collaborative robotic arms are used in almost all spheres of industry because in addition to classic industrial precision, they also provide safety and protection, so that people can move freely in a collaborative environment.

Direction in which this project is developing, in its primary focus has demonstration that will with the help of new 3D scanning technologies, enable robots to perform functions with great precision. This represents the something that man cannot perform with his free hand. An example of such functions is robotic surgery.

III. 3D SCANNER HARDWARE DESCRIPTION

The system is primarily composed of Logitech C615 web camera (maximum resolution 1920 x 1080 pixels, and range from 30fps to 5fps), Projector - Acer K132 (resolution 1280 x 800 and 500 ANSI Lumens), Structured Light scanning software and calibration pattern and calibration panel. The angle between the walls of the calibration panel is 90 degrees and the size of the calibration pattern depends on the size of the object being scanned.

Additional hardware components required for construction of automatic 360° scanning system for rotating platform, Kinco 2S42Q-0240 Step motor, Kinco 2M412 stepper motor driver, DC Power supply, Arduino UNO, HDMI and USB cables. Algorithm code in Arduino software allows communication

between Structured Light scanning software and rotating platform. The optimal angle of the camera in relation to the projector is 22-25º. For objects larger than 350mm the camera is placed to the right of the projector.

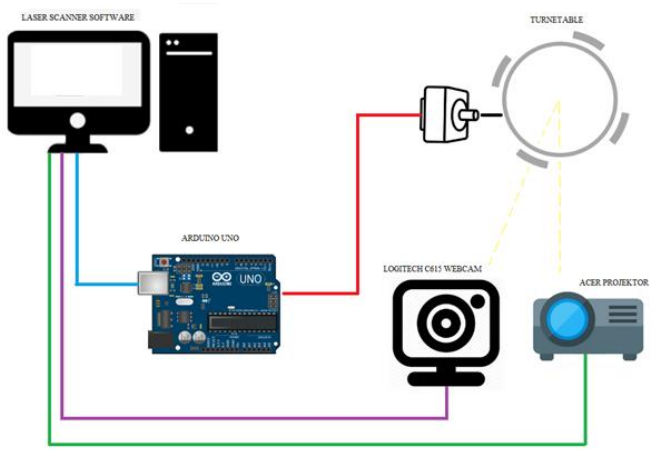

Fig. 1. Hardware configuration of the automatic 3D scanner

IV. DESCRIPTION OF THE APPLICATION INTERFACE

The calibration process consists of two key parts, camera calibration and projector calibration. In order to obtain a projection matrix from the world coordinates to the image in the process of calibrating the camera, the calibration pattern must have at least six points that are clearly visible.[6]

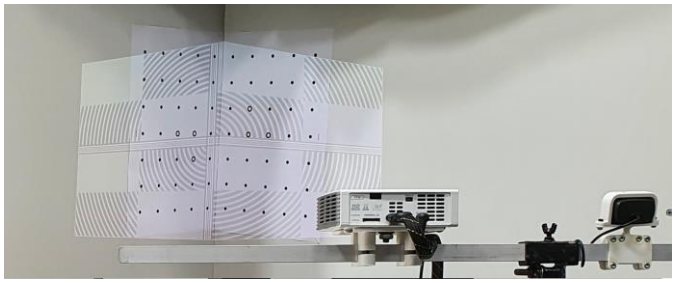

Fig. 2. Calibration process

The projector illuminates a striped pattern, so that software can determine the relative position of the camera and projector. The calibration template has many points, so that in the calibration process the projector can estimate all the unknown coefficients, unambiguously.

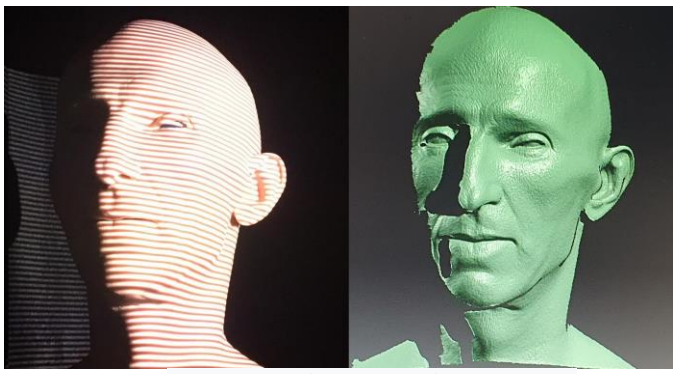

Fig. 3. Scan process (left) and result of one scan (right)

The software has the ability to communicate via the com port by sending numbers and letters. In advanced settings, it is possible to personalize the message that is sent and received.

The commands that the software receives are (Commands):

- Q – AddToList
- I – StartScan
- U – ModeSL
- P – StopLaserScan

The characters that the software sends are (Messages):

- 0 – StartSLScan
- H – GrabImage
- C – ModeSL

An algorithm implemented in Arduino that allows synchronizing movement of the rotating platform and the scanning software:

Serial.available() function returns number of bytes(characters) available for reading from serial port. Input is placed in variable c and the program waits for c to have a value of 0 which represents that the scanning has started. After that, the next character that arrives from the serial port is entered in the variable p, as long as p does not get the value of H, which means that the frame is captured, it is reloaded. When p is equal to H, the algorithm waits for msAfterPictureTaken. The value of this time content depends on the complexity of the scan, the image quality, and whether it is necessary to collect the texture color. A Q character is then sent to the serial port, which is a command to the software to save the current scan in a List. Program then waits for msAfterSave, and the platform on which the object is located rotates by Rotation angle. The Rotation angle value is determined before the scan begins and depends on the complexity of the object. In order for the platform to stabilize after the change of position, it is necessary to wait for msAfterRotation, the cnt counter is incremented by one and the U command is sent to the software to prepare for further scanning, to adjust all camera and projector parameters. If the value of the cnt counter is less than the number n obtained when 360 is divided by the Rotation angle, full rotation is not completed and scanning is continued by sending an I character to the serial port indicating the command for the software to start scanning. If a full rotation is performed, the command P is sent the scanning process is stopped and thus the process of collecting individual scans is completed.

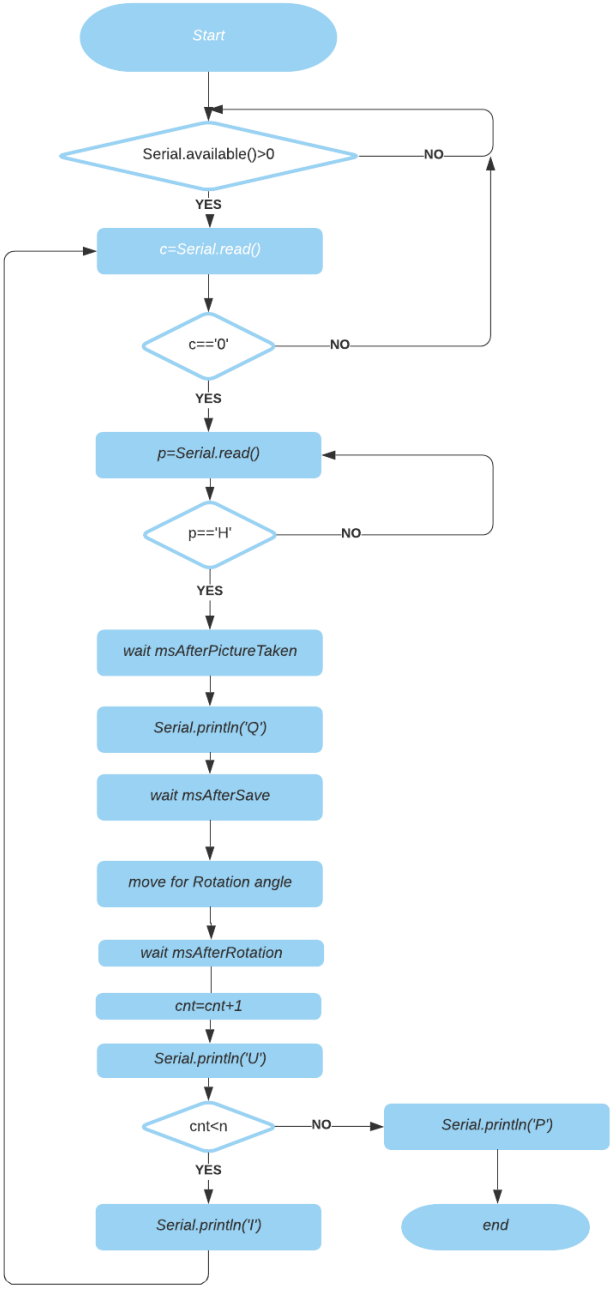

Fig. 4. Flowchart of the algorithm for synchronizing movement of the rotating platform and the scanning software

The duration of the scanning process depends on the quality of the scanned object required for proper application, so it needs to be determined experimentally. After collecting N scans, they are saved and exported to MeshLab to be aligned and then fused.

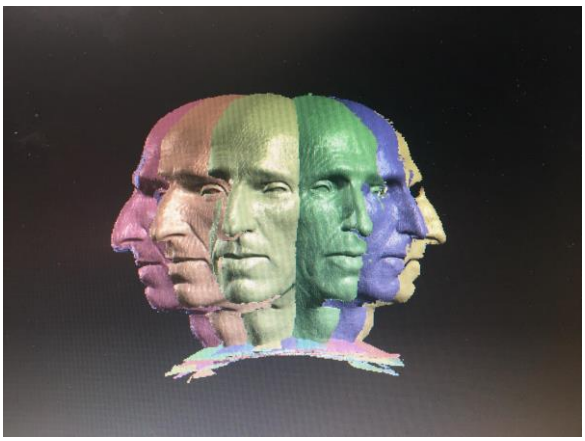

Fig.5. Screenshots of the succesive scans

The result of the scan is the obj file format – it represents 3D geometry of the object. File contains:

- v-vertices 3xN matrix contains x, y, z coordinates of points.
- vt- texture vertices optional if texture capture is enabled during scanning.
- vn- vertex normal.
- f- faces (face is any of the individual flat surfaces of a solid object)

For files of this complexity to be used in the simulation in MATLAB, it is necessary to simplify the scan result and for that the MeshLab software package was used.

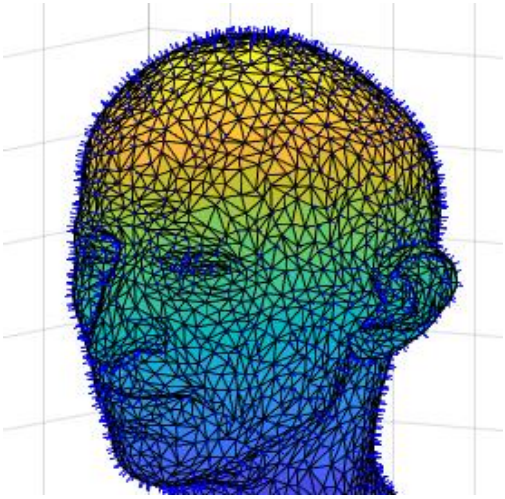

Fig.6. Mesh representation – of simplified scan in MATLAB

The position vectors consist of the x, y and z coordinates of each vertex. Let n_α be the normal vector derived from the mesh file, the components of the rotation matrix of the end effector are obtained as follows [8]:

$$\hat{a} = -\frac{n_a}{\|n_a\|}, \hat{n} = \hat{a} \times \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \hat{o} = \hat{a} \times \hat{n} \quad (1)$$

$$R_{EE} = \begin{bmatrix} \hat{n} & \hat{o} & \hat{a} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (2)$$

From there we can derive values that represent the orientation of each vertex:

$$\begin{aligned} \psi &= atan2(r_{21}, r_{11}) \\ \theta &= atan2(-r_{31}, cos(\psi)r_{11} + sin(\psi)r_{21}) \\ \varphi &= atan2(sin(\psi)r_{13} - cos(\psi)r_{23}, -sin(\psi)r_{12} + cos(\psi)r_{22}) \end{aligned} \quad (3)$$

V. Connection of Scanner and Robot Controller

Universal robots have the ability to communicate over TCP/IP protocols (TCP socket connection over Ethernet). A PC is a server while a robot is a client. It is possible to establish communication by writing scripts in different programming languages, but as the simulation was performed in MATLAB the communication is also implemented in MATLAB. MATLAB has built-in functions for creating servers and clients. It is necessary to know the IP address and port number. Server waits for client connection. The robot needs to send the message that the server expects in order to get the first desired waypoint as feedback.

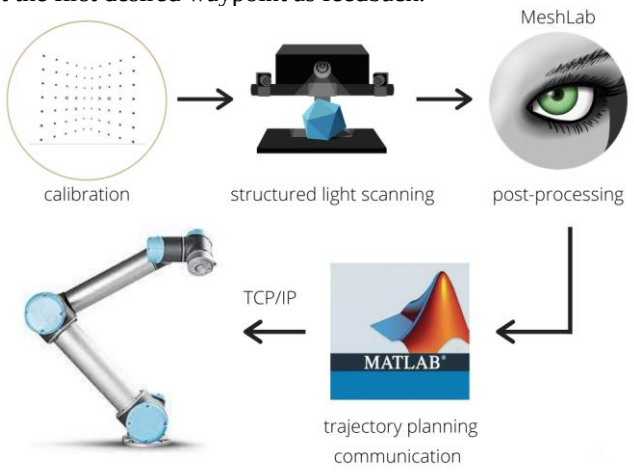

Fig.7. High-level description of the integrated system - 3D scanner and industrial robot UR-5

The message the robot receives consists of the number of data sent and that data, so that at any time it is possible to check whether the message has been sent in its entirety (format in PolyScope: *receive_data:=[6,0,0,0,0,0,0]*).

Data sent from server must be in bracket, each data must be followed by a comma unless it is the last, at the end there is '/n'. The robot has three main ways of calculating how to move from Waypoint to Waypoint which is a nonlinear movement “MoveJ”, a linear movement, “MoveL” and a circular movement (MoveC) which is under a Process move “MoveP”. [9]

The Universal Robot is controlled on Script Level, URScript is the robot programming language (PolyScope software).

An example of the functions used [10]:

- *Movel(pose, a=acceleration, v=speed,r=blend radius)*
  move to position (linear in tool-space), a[m/s2], v[m/s], r[m].
- *get_forward_kin()*
  returns tool pose-forward kinematic transformation (joint space to tool space) of current joint positions.
- *get_inverse_kin(x)*
  returns joint position- Inverse kinematic transformation (tool space to joint space). Solution closest to current joint positions is returned.
- *socket_open(server, port)*
  Open ethernet communication.
- *socket_read_ascii float(number)*
  Reads a number of ascii float from the TCP/IP connected.
- *socket_send_string(str)*
  Sends a string to the server Sends the string through the socket in ASCII coding

Robot sends current tool positions (fun. *socket_send_string*) in the format: p[x,y,z,ψ,θ,φ]. Functions for reading current positions of the final device are written in MATLAB.

VI. Demonstration Example

Experimental verification of the system was performed by scanning the model of Nikola Tesla's head.

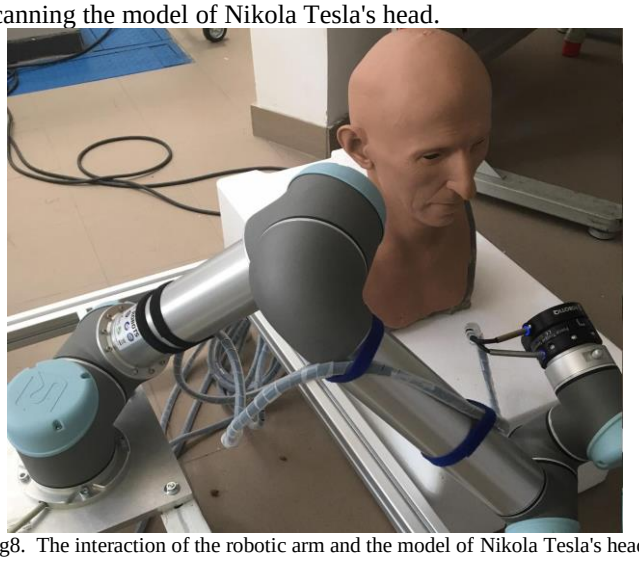

Fig8. The interaction of the robotic arm and the model of Nikola Tesla's head

In order to better simulate the movement of the robotic arm, the points obtained as a cross section of the point clouds and horizontal plane passing through the axis of symmetry were chosen as a demonstration example.

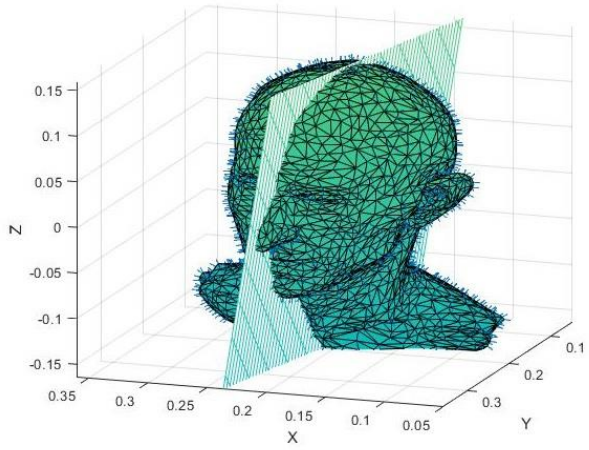

Fig.9. Intersection of planes and mesh representations

The distance of each point from the plane was calculated and based on that, points with a smaller distance from the threshold were selected. A value of 5mm was chosen for the threshold and *Figure10*. shows which points meet this criterion.

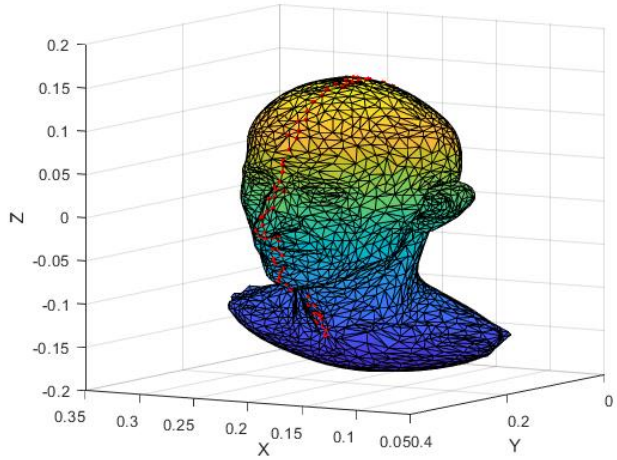

Fig10. Selected points

The robot from the initial position crosses the contour of the face at a constant speed of one cm per second, continuously without stopping.

A laser is placed on the end effector for easier visualization. In order to find a point on the line(*X*) containing the normal vector(*n*) that passes through the original point(*P*) on the scanned head model and is *d*=1 cm away from the original point, the following system of equations is solved:

$$\overrightarrow{n} = \begin{bmatrix} l \\ m \\ s \end{bmatrix} \qquad \begin{matrix} P = (x_1, y_1, z_1) \\ X = (x, y, z) \end{matrix} \tag{4}$$

$$t = \frac{x - x_1}{l} = \frac{y - y_1}{m} = \frac{z - z_1}{s} \tag{5}$$

$$d = \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2} \tag{6}$$

The result of the displacement of the profile points by 1 cm in the direction of the normal is shown in the *Figure11*.

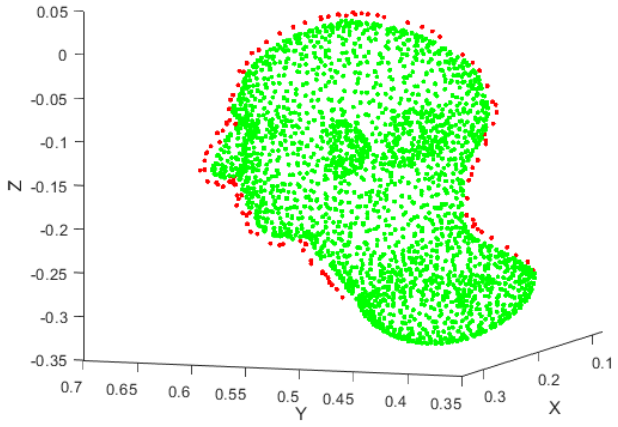

Fig.11. New selected points

For the purposes of simulation, the UR5 robot was imported from the Robotics System Toolbox, as rigid Body Tree.

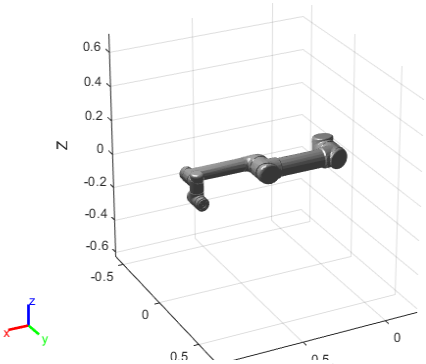

Fig.12. Robot UR5 in home configuration position

By calling the *inverseKinematics* function, an inverse kinematics object is formed which, based on the desired position and orientation of an end-effector computes the joint configuration that leads to it.[11] The numerical optimization used in the calculation of inverse kinematics is the Broyden – Fletcher – Goldfarb – Shanno algorithm (BFGS):

Taking into account the starting point(q_0) and the initial Hessian matrix(H_0) for each iteration(k), the search direction(d_k) is calculated:

$$d_k = -H_k g_k \tag{7}$$

If the gradient(g_k) is zero, the search stops. If not, step size (α_k) is calculated with line search method, and new point is obtained:

$$x_{k+1} = x_k + \alpha_k d_k \tag{8}$$

Then the new Hessian matrix is recalculated, and the stop criterion is examined. If it is not fulfilled, it moves to the next iteration.[12]

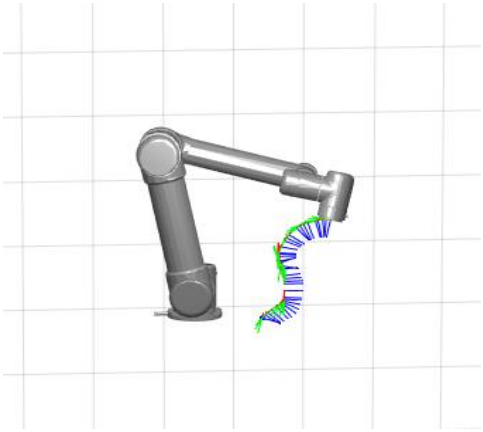

Fig.13. Simulation of trajectory monitoring

Motion trajectory was implemented as Joint space and Task space trajectory. In addition to interpolating the positions between each waypoint, it is necessary to implement interpolation between orientations. This is achieved by interpolation between quaternions owning to the fact that if we interpolated the Euler angles the solution would not be unique. The way this is implemented in MATLAB is the SLERP (Spherical Linear Interpolation) method.

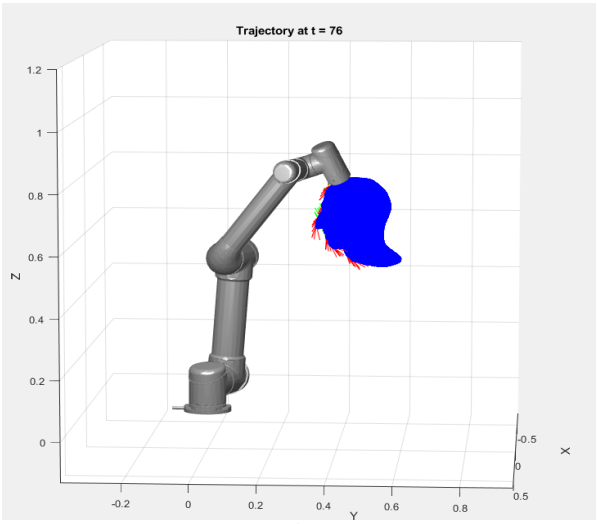

Fig.14. Simulation of trajectory monitoring

Another way to obtain the internal angles of the robot from the external coordinates of position and orientation is to implement an analytical procedure for solving inverted kinematics.[13] As there are several configurations that can produce a certain position and orientation, one of the solutions is to choose the configuration that makes the smallest change in the position of the joints compared to the previous point. This method is much slower, so numerical calculation is applied with the help of libraries and built-in functions in the MATLAB Robotics Toolbox.

VII. Conclusion

Automatic 3D object scanning facilitates processes not only in robotics but also in other important spheres of life. Collaborative robots and their integration with such systems provide a new form of technological advancement that aims to improve the way people live and work. The results of the scanning of Nikola Tesla's head met the criteria of precision. The goal of this work, the construction of automated scanning and connection with the universal robot UR5 is achieved. Automated scanning is implemented using simple hardware for the motorized movement, and low cost or open-source software. Moreover, the process of filtering and collecting point clouds that form the robot trajectory is described. Experimental verification of the system was performed and communication is established.

Acknowledgment

The results presented in the paper are obtained in the scope of the research projects: „Development and Experimental Performance Verification of Mobile Dual-Arms Robot for Collaborative Work with Humans “, Science and Development Programme – Joint Funding of R&D Projects of the Republic of Serbia and the People's Republic of China, contract no. 401-00-00589/2018-09, 2018-2021 and national R&D project no. TR-35003, both supported by the Ministry of education, science and technology development of Republic Serbia.

References

[1] O. Vozisova, S. Eroshenko, E.Koksharova, A. Khalyasmaa and S. Dmitrir (2016, March), “Application of 3D Scanning and Printing Technologies in Electric Power Industry,” Ekaterinburg, Russia. In 2016 IEEE International Conference on Industrial Technology (ICIT) (pp. 892-897). IEEE

[2] S. Koceski, N. Koceska, P. B. Zobel and F. Durante, "Characterization and modeling of a 3D scanner for mobile robot navigation," 2009 17th Mediterranean Conference on Control and Automation, 2009

[3] E. Digor, A. Birk, A. Nüchter “Exploration Strategies for a Robot with a Continously Rotating 3D Scanner”, Simulation, Modeling, and Programming for Autonomous Robots, 2010, Volume 6472

[4] Ogorodnikova, O., Olchanski, D. “On a 3D scanning robot system design problem”, Periodica Polytechnica Mechanical Engineering, 51(1), pp. 39–44, https://doi.org/10.3311/pp.me.2007-1.06

[5] Etienne Dombre „Introduction to Surgical Robotics“, LIRMM, Montpellier 2009.

[6] Josep Forest i Collado, “New methods for triangulation-based shape acquisition using laser scanners,” *Girona, maig 2004*.

[7] Du Q. Huynh, “Calibration of a Structured Light System: A Projective Approach,” Centre for Sensor Signal and Information Processing Signal Processing Research Institute Technology Park, The Levels, SA 5095.

[8] Jelena Z. Vidaković, Vladimir M. Kvrgić, Mihailo P. Lazarević, Zoran Z. Dimić, Stefan M. Mitrović (2017), “Procedure for Definition of End-effector Orientation in Planar Surfaces Robot Applications,” Belgrade Tehnika, 72(6), 845-851.

[9] UNIVERSAL ROBOTS User Manual, Version 3.3.3.

[10] The URScript Programming Language Manual.

[11] Peter Corke, Robotics Toolbox for MATLAB, Release 9.10, February 2015. https://petercorke.com/toolboxes/robotics-toolbox/

[12] Atikah Ramli, Ibrahim Jusoh, Mohd Rivaie Mohd Ali (2017), “A combination of Broyden-Fletcher-Goldfarb-Shanno (BFGS) and n-th section method for solving small-scale unconstrained optimization”, Malaysian Journal of Fundamental and Applied Sciences, 13(4), 717-720.

[13] Ryan Keating, “UR5 Inverse Kinematics”,Johns Hopkins University.