Simulation and Programming of an Industrial Robot Based on Augmented Reality

https://doi.org/10.31713/MCIT.2019.59

Vitalii Kutia
Aeolus Robotics Inc.
Wroclaw, Poland
vitalii.kutia@pwr.edu.pl

Filip Ruchel, Krzysztof Chrapek
Faculty of Mechanical Engineering
Wroclaw University of Science and Technology
Wroclaw, Poland

Abstract— **This work is devoted to design, development and practical implementation of an AR application for simulation and programming of an industrial robot basic movements. The process of the AR-based application for mobile devices is briefly described to show the advantages and scalability of the proposed concept.**

Keywords— augmented reality; industrial robot; inverse kinematics; intuitive robot programming; human-robot interaction

I. INTRODUCTION

The factor that has a significant impact on the progress in robotization of industrial and production processes is a growing intelligence of technical solutions at robotic stations, although there are still no fully autonomous workstations in industry. Growing customer requirements and global competition are factors that cause global industrial leaders to apply industrial robots and solutions of Industry 4.0 increasingly. It is noted in [1] that fast deployment of industrial robotic cells requires the improvement of industrial robot programming process to make it shorter, easier, cost-effective and user friendly. The use of virtual reality (VR) and augmented reality (AR) technologies is one of the most promising techniques of intuitive robot programing that has been developing at a rapid pace. These techniques enable to program industrial robots by persons without special programming skills in more intuitive way which can significantly reduce the costs and time for integration of industrial robots in manufacturing enterprises.

As pointed out in [2], the market and use of AR is growing rapidly. In the automotive, aviation and advanced information technologies industries, solutions based on augmented reality have been successfully introduced. AR on the robotics market has a chance to create a user-friendly environment that facilitates programming of robots. Guided by this thought, an application was created that is the subject of this work.

I.APPLICATION DESIGN AND IMPLEMENTATION

A. *Application Development Algorithm*

The designed application should be intuitive, easy to deploy and use on mobile devices and should allow for programming of simple movements of virtual robot. The application will be made in such a way that its functionality resemble an IR teaching pendant. The main task of the application is to impose a virtual model of the robot arm on the real image captured by camera in a tablet PC or smartphone with Android OS. The virtual robot model should keep the dimensions and all technical features of the real robot arm.

In this work, the next development procedure of the AR-based application was implemented:

- selection of an IR model, development of the application hardware and software architecture;
- design of 3D model of the selected robot;
- implementation of the IR inverse kinematics solver;
- generation of markers;
- integration of the robot 3D model into AR software;
- registration of the virtual robot model in the image of real environment;
- simulation of the virtual robot movements;
- saving of TCP coordinates for robot program.

B. *Hardware and Software Description*

In accordance to the algorithm described in previous section, the industrial robot Mitsubishi Melfa RV-3AL has been selected as a prototype for virtual 3D model to be simulated in the application. The initial version of the 3D model was created in Autodesk Inventor (Fig. 1). Then each part of the model was exported to Blender software to get the whole robot model in the FBX format.

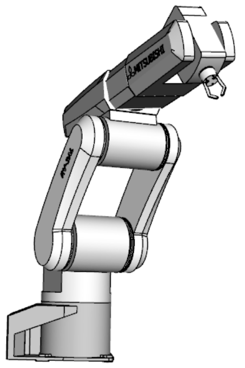


Figure 1. 3D model of the Mitsubishi Melfa RV-3AL robot arm

The very initial purpose was to create an application for mobile devices. The target system is Android OS platform. The hardware basis for application is the tablet PC Samsung Galaxy Tab S4. The main advantages of this device are: 8-core processor Snapdragon 835, 4 GB RAM, 10.5” Super AMOLED screen with 2560x1600 px resolution, 13 MP rear camera and Android 8.1 version installed.

One of the most important elements of this work is to select the appropriate software that will allow develop the target application. The required software can be divided into two parts. The first part is a graphics engine that supports 3D models and physics. The second necessary part is a dedicated additional library for AR-based applications development.

In order to choose the appropriate programming interface, OpenGL and Unity 3D were taken into account. OpenGL is a multi-platform graphical interface enabling the creation of advanced 2D and 3D graphics in many programming languages. It is mainly used to create computer games and data visualization. Supported languages include C, C ++ and Java. Unity 3D is an integrated environment that allows development of 2D and 3D graphics on most available operating systems. Working in Unity, you can implement the application for Windows, Mac, Linux, Android, iOS platforms and for popular game consoles. Supported programming languages are C#, Boo and UnityScript. Unity works with two libraries for AR, namely ARCore and Vuforia [3]. An important advantage of this environment is the extensive graphical interface of the program. It enables the efficient development and testing of applications and can significantly speed up the effects of work.

Unity 3D and Vuforia SDK have been selected to create the AR-based application. The source code of the application is written in C# in the Visual Studio environment.

C. Forward and Inverse Kinematics Solution

The basis for robot control is the solution of forward and inverse kinematics problems. The forward kinematics problem is to calculate the position and rotation of the end-effector relative to the base reference system by specifying the joint parameters. The inverse kinematics problem consists in calculating the joint parameters from specified position and orientation of the end-effector.

In both of these solutions, the Denavit-Hartenberg notation was used to describe the robot manipulator. The Denavit-Hartenberg parameters are:

- a_i – distance between the axes z_{i-1} and z along the x_i axis,
- α_i – angle between the axes z_{i-1} and z around the x_i axis,
- d_i – distance between the x_{i-1} and x_i axes along the z_{i-1} axis,
- θ_i – the angle between the x_{i-1} and x_i axes around the z_{i-1} axis.

The Denavit-Hartenberg parameters for the Mitsubishi Melfa RV-3AL manipulator with 6 degrees of freedom are presented in Table I.

Unity 3D software enables to solve the forward kinematics problem using the hierarchy of objects.

TABLE I. THE DENAVIT-HARTENBERG PARAMETERS OF THE MITSUBISHI RV-3AL MANIPULATOR

i	*a*	*α*	*d*	*θ*
1	a_1	$\pi/2$	d_1	θ_1
2	a_2	0	0	θ_2
3	a_3	$\pi/2$	0	θ_3
4	0	$-\pi/2$	d_4	θ_4
5	0	$\pi/2$	0	θ_5
6	0	0	d_6	θ_6

In general, the solution of an inverse kinematics problem can be found in three ways: matrix, vector and numerical methods. The matrix method uses the equations of transformation matrices written in Denavit-Hartenberg notation. The vector method uses an equation with three unit vectors, and the numerical method uses algorithms, such as Newton-Raphson method [4]. In our case, the solution of the inverse kinematics was found using the matrix method.

This problem comes down to find components of the vector q:

$$q = [\theta_1\ \theta_2\ \theta_3\ \theta_4\ \theta_5\ \theta_6]^T, \tag{1}$$

while transformation of ${}^0T_6$ is a function of variables $\theta_1, \theta_2, \ldots \theta_6$

$${}^0T_6 = {}^0A_1(\theta_1)\cdot{}^1A_2(\theta_2)\cdot{}^2A_3(\theta_3)\cdot{}^3A_4(\theta_4)\cdot{}^4A_5(\theta_5)\cdot{}^5A_6(\theta_6). \tag{2}$$

In the case of inverse kinematics, the left side of equation (2) is known and takes the form of a 4 x 4 matrix

$${}^0T_6 = \begin{bmatrix} & B & & p \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where

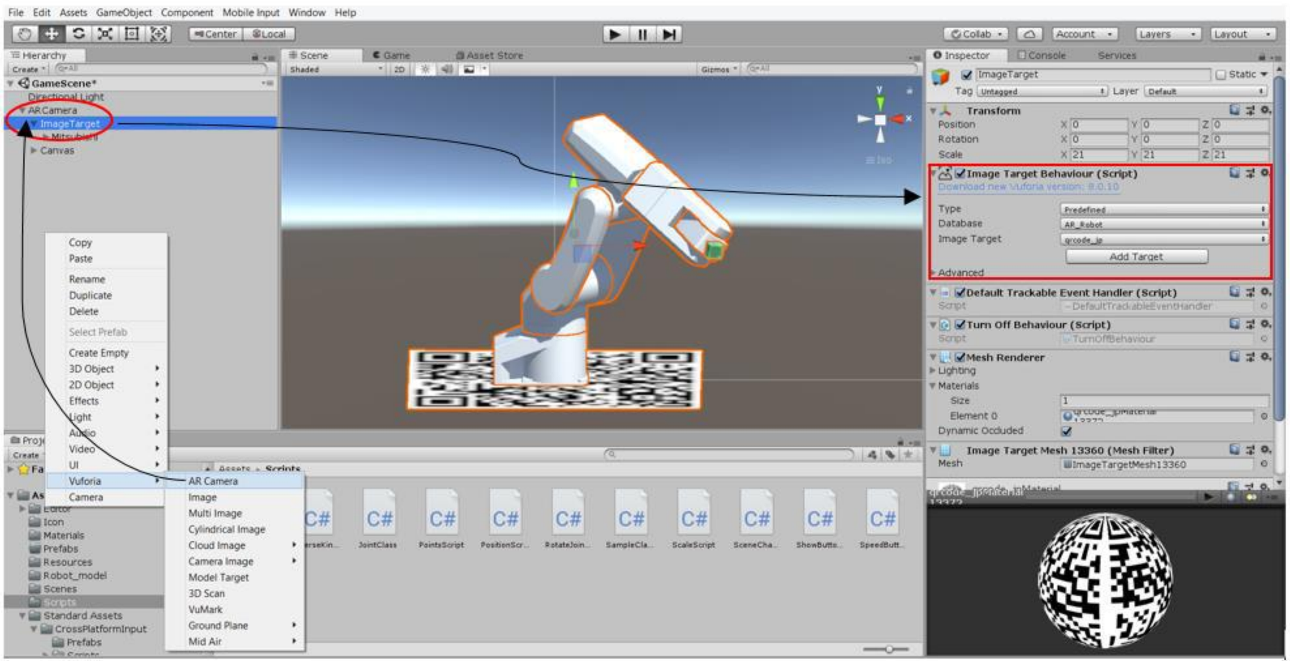
$$B = \begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix}, \text{ and } p = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}. \tag{4}$$

Matrix B is a rotation matrix, while matrix p is an effector position matrix. On the other hand, the right side of the equation (2) is multiplied transformation matrices, written according to the formula:

$$A_{i,i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & -\cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

The analytical solution of the equation (2) was found and implemented as C# script in Unity 3D IDE.

Thanks to the implemented forward and inverse kinematics, an operator is able to control the arm directly by changing the rotation angle of individual axes or by setting the end-effector position.



ImageTarget
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 21 Y 21 Z 21
Image Target Behaviour (Script)
Download new Vuforia version: 8.0.10
Type Predefined
Database AR_Robot
Image Target qrcode_jp
Add Target
Advanced
Default Trackable Event Handler (Script)
Script DefaultTrackableEventHandler
Turn Off Behaviour (Script)
Script TurnOffBehaviour
Mesh Renderer
Lighting
Materials
Size 1
Element 0 qrcode_jpMaterial
Dynamic Occluded ✓
Image Target Mesh 13360 (Mesh Filter)
Mesh ImageTargetMesh13360

Figure 2. The process of integration the robot 3D model in Unity 3D with Vuforia SDK

D. The Robot Model Visualization in Augmented Reality

The Vuforia SDK integrated into the Unity 3D environment has been used to visualize the 3D model of the industrial robot in AR. The Vuforia platform make it possible to generate 3D objects on a real image using so-called markers. The marker can be associated with characteristic graphics in the form of a QR code. However, any graphic that retains some of the features of diversity can also be a marker. Vuforia has an internet platform where developers can create marker databases and add selected graphics. Then such database is processed and can be imported into Unity 3D (Fig.2).

When the application is launched and the required marker is clearly visible for the camera device, the interactive model of the virtual robot is registered and appears on the image of the real environment (Fig. 3).

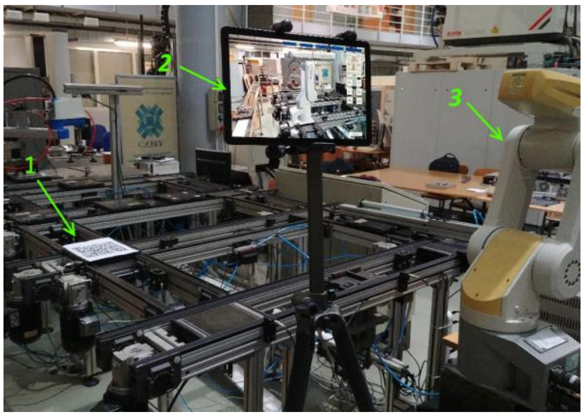


Figure 3. Tests of the developed application: 1 – marker; 2 – the tablet PC with the developed AR application; 3 – real robot arm

The screencast video [5] from the tablet with working application shows main features and functionality implemented in the application.

II. Conclusions

Summing up the entire application development process, it can be stated that the basic design assumptions have been met. The AR-based application for the IR simulation and programming has been successfully created according to the step-by-step procedure presented in the Section II. As a result of this work, a user-friendly application for mobile devices has been created that allows simulation of simple robot movements in AR. Regarding the AR or VR based techniques, there are many indications that this is a future direction of development, which offers unprecedented opportunities in different fields of engineering. The prospect of IR intuitive control by ordinary mobile devices without the need for a highly qualified programmers seems to be possible in the nearest future.

References

[1] K. Krot and V. Kutia, “Intuitive methods of industrial robot programming in advanced manufacturing systems,” Intelligent Systems in Production Engineering and Maintenance ISPEM-2019, 2019, pp. 205-215.

[2] M. Campbell, D. Busiek, J. Lang, “The State of Industrial Augmented Reality: A Spotlight on Industrial Innovation”, PTC, 2018, https://www.ptc.com/-/media/Files/PDFs/Augmented-Reality/State-of-AR-Whitepaper2.pdf

[3] Unity features. https://unity3d.com/unity/features/multiplatform Accessed 01.09.2019.

[4] A. Morecki, J. Knapczyk. Basics of robotics: theory and elements of manipulators (in Polish), Warsaw: WNT, 1999, p. 78.

[5] Video representing application functionality: https://youtu.be/5H1eA3DPeRQ