Kinematic and Dynamic Analysis and Simulation of Complex Mechanisms Using the MATLAB Software

Ivan Milićević^{1*}, Vojislav Vujičić¹, Radomir Slavković¹, Nedeljko Dučić¹, Marko Popović¹

¹ Faculty of technical science, Čačak, University of Kragujevac, Serbia

The research presented in this paper is related to the development of complex mechanisms using virtual models obtained by integration various modern computer programs, specialized in certain fields of computer science. The industrial robots, as a typical representative of complex mechanisms with six degrees of freedom, carried out the definition, analysis and simulation for the considered mechanism. Virtual model with a control algorithm implemented in the MATLAB, as an open kinematic chain. This software solves the problem with difficult matrix algebra and by using the add-on Robotics Toolbox can be made behavioural simulation, and verification for the model which provides analysis of the behaviour for real robots. In this case we use industrial robot FANUC S-430iF, where we wrote up the whole process for solving kinematics and dynamics for complex mechanisms by using computers, from the formation of the transformation matrix through solving direct and inverse kinematics and dynamics, as well as movement simulation. To define the 3D model was used SolidWorks.

Key words: mechanism, analysis, simulation.

1. INTRODUCTION

Computing development in recent years has enabled the creation of numerous software that can efficiently solve various engineering problems. However, the modern principle, integrated design approach [1] (Fig. 1), demands for a comprehensive analysis in order to obtain satisfactory results, often makes it necessary to carry out the integration of modern software specialized for certain areas of the design into a program unit. Thus, for example, in the complex mechanisms design [2-4], it is necessary to use some of the CAD software which can be used to make geometric 3D models, with software that can simulate movement, analyze kinematics and dynamics for the moving parts in the mechanisms [5], as well as design management through appropriate control algorithms.

Fig. 1: Integrated design approach

Matlab is widely applicable multidisciplinary engineering software which enables mathematical computations, developing control algorithms, simulation and process analysis, data processing, visualization, and all this through interactive and programming work. [6-8]

Simulink, as an integral part of the *Matlab* is used to simulate the dynamic model (the graphical environment). It can analyze the linear, nonlinear, time-continuous and/or discrete time models with multiple inputs and outputs and with lumped parameters. Simulink is used for forming a control algorithm, simulation and analysis models [9]. It is a visual tool that enables the simulation of continuous and discrete systems using functional block diagrams and thus does not require the user detailed knowledge of the syntax of a programming language.

Robotics Toolbox [10] is a software add-on for *Matlab* specializes in analysis and simulation of industrial robots. It's free, but it is part of *Matlab*. This toolbox allows many functions that are useful in robotics, including estimate of trajectory, kinematics, dynamics and movement simulation. It is based on the theory of presenting manipulator as an open kinematic chain consisting of rigid segments interconnected joints (connections), with use of standard Denavit-Hartenberg notation. [11]

Geometry of robot was generated in **SolidWorks** [12-14]. 3D models of individual components were saved as .stl files (*StereoLithography* or *Standard Tessellation Language* - one of the standard formats used to describe the geometry of three-dimensional object) imported into the *Matlab* environment.

2. MODEL DEFINING

2.1. Input parameters

In order to be able to use the software, first of all necessary to define the input data, i.e. mass and geometric characteristics of the model. Some of the necessary data

are taken from the website of the manufacturer of robots used as example in this study. Those data are shown in Table 1:

Table 1: Mass and geometric characteristics

Axis	Segment mass [kg]	Segments center of mass position relative to its coordinate system [m]		
n	m	r_x	r_y	r_z
1	116.2585	0.02714	0.53479	0.08050
2	60.79505	0.26144	1.21097	0.34386
3	64.47270	0.01999	1.95873	0.50694
4	6.62643	0.03656	2.06500	1.48849
5	5.12291	0.01912	2.06500	1.67359
6	0.72895	0.00875	2.06500	1.79944

Axis	Segments tensor of inertia around its center of mas [Kg*m ²]					
n	L_{xx}	L_{yy}	L_{zz}	L_{xy}	L_{xz}	L_{yz}
1	9.12674	8.61229	8.49437	0.16098	-0.57145	1.65126
2	8.19389	0.76659	8.02259	0.42771	0.00337	0.05880
3	6.62158	6.06231	2.08597	-0.10243	-0.02078	0.89298
4	0.08972	0.09288	0.02856	0.00000	0.01767	0.00000
5	0.03819	0.06668	0.04821	0.00000	-0.00191	0.00000
6	0.00130	0.00131	0.00243	0.00000	0.00000	0.00000

Also, it is necessary to determine Denavit-Hartenberg's (D-H) parameters. D-H notation is assigning spatial coordinate system for different joints of manipulator. In Fig. 2, the segment i connecting the segment $i-1$ to the segment $i+1$. The method involves determining the four parameters needed to obtain complete homogenous transformation matrix. [15-18] These parameters are:

- link length or segment a - the shortest distance between the axis of the joints Z_{i-1} and Z_i (the distance from the intersection axis Z_{i-1} with the axis X_i to coordinate the start of the i -th coordinate system, measured along the X axis;
- rotation angle α - the angle between the axis of the joints Z_{i-1} and Z_i measured in the plane perpendicular to a_i (the angle of rotation around the X axis, you need to axis lead Z_{i-1} in a parallel position to the axis Z_i , by using the right hand rule).
- The distance between the segments (offset) d_i - distance from the beginning of the $i-1$ -th coordinate system to the intersection axis Z_{i-1} with the X axis, measured along the axis axis Z_{i-1} .
- Joint angle θ - the angle of rotation around Z_{i-1} axis, you need to lead axis X_{i-1} in a parallel position to the axis X_i (using the right hand rule).

The parameters a_i and α_i are defined by geometry of manipulators and have a constant value (depending on whether the joint is revolute or prismatic) [15-18], while the parameters d_i and θ_i can be variable.

Fig. 2: Schematic of the adjacent axes with the appropriately assigned reference frames for determining the D-H parameters

The following steps describe the systematic determination of the D-H parameters:

1. Mark each segment axis with number, starting at 1 (axis of base) to n (axis of end-effector). O_0 is the origin of the base coordinate system. Each joint must be assigned with axes.
2. Set the coordinate system for each joint, according to the rule of the right hand. Start with the base joint. For rotational joints, the axis of rotation i is always in the direction Z_{i-1} . If it is a translatory joint, Z_{i-1} should be placed in the direction of translation.
3. Set the X_i axis. X_i axis direction is obtained from the expression $\mathbf{x}_i = \pm \frac{\mathbf{z}_{i-1} \times \mathbf{z}_i}{\|\mathbf{z}_{i-1} \times \mathbf{z}_i\|}$ or along the common normal between the axis Z_{i-1} and Z_i when they are parallel.
4. Axis Y_i should be directed to give a rectangular coordinate system to the right hand rule
5. For the next joint, if this is not the last joint, repeat steps 2-4.
6. To end-effector (gripper) axis Z_n have to be positioned in the direction in which the handle is approaching.
7. Angle joint θ_i the angle of rotation around Z_{i-1} axis, so that the axis X_{i-1} lead in parallel position to the axis X_i .
8. Rotation angle α_i is the angle of rotation around the X axis, you need to lead axis Z_{i-1} in a parallel position to the axis Z_i .
9. Link length a_i is perpendicular distance between the axis i and axis, $i+1$.
10. The distance between the segments (offset) d_i is the distance along the axis Z_{i-1} , as shown in Fig. 6. It is, therefore, the distance between points O_{i-1} and O_i along the Z_{i-1} axis (the axis i).

The following illustration shows the determination of the D-H parameters for the selected robot model *FANUC S-430iF*, on the basis of the procedures described above, and the results are shown in Table 2. [19]

Fig. 3: Marking axis and rotation of axis

Fig. 6: Link length and distance between segments

Table 2: D-H parameters

Axis	The distance between segments d [m]	Link length a [m]	Rotation angle α [°]
1	0,514	0,305	$-\pi/2$
2	0	1,075	$-\pi$
3	0	0,250	$-\pi/2$
4	1,275	0	$\pi/2$
5	0	0	$\pi/2$
6	0,240	0	0

Fig. 4: Defining coordinate systems for each axis

Fig. 5: Positioning angle for each axis

3. MODELING OF FANUC S-430IF MANIPULATOR

Modeling of manipulator using MATLAB contains two parts:

- Creating a script
- Creating Simulink model

3.1. Creating a script

Creating a script involves the importation of D-H parameters, as well as mass and geometric characteristics of the model. Since the script doesn't exist, it is necessary to create a new script in the folder with active path (Fig. 7).

Fig. 7: Creating a script

After creating and recording the script code is entered in the script. Shown below is the contents of the script (code) which provides for the creation of the object in MATLAB based on the entered model characteristics.

```
%robot Fanuc S-430iF
```

```

%robot limits
THETA1_MAX=deg2rad(360);
THETA2_MAX=deg2rad(134);
THETA3_MAX=deg2rad(362);
THETA4_MAX=deg2rad(720);
THETA5_MAX=deg2rad(250);
THETA6_MAX=deg2rad(720);
%definisanje D-H parametara
clear L
L(1)=Link('revolute','d',0.514
,'a',0.305,'alpha',-1.571);
L(2)=Link('revolute','d',0,'a',1.075,'alpha',-
3.142);
L(3)=Link('revolute','d',0,'a',0.25,'alpha',-
1.571);
L(4)=Link('revolute','d',-1.275
,'a',0,'alpha',1.571);
L(5)=Link('revolute','d',0,'a',0,'alpha',1.571);
L(6)=Link('revolute','d',0.24,'a',0,'alpha',0);
%definisanje mase segenata [Kg]
L(1).m=116.25857;
L(2).m=60.79505;
L(3).m=64.47270;
L(4).m=6.62643;
L(5).m=5.12291;
L(6).m=0.72895;
%polozaj centra mase u odnosu na njegov
koordinatni sistem [x y z]
L(1).r=[0.02714 0.53479 0.08050];
L(2).r=[0.26144 1.21097 0.34386];
L(3).r=[0.01999 1.95873 0.50694];
L(4).r=[0.03656 2.06500 1.48849];
L(5).r=[0.01912 2.06500 1.67359];
L(6).r=[0.00875 2.06500 1.79944];
%elementi tenzora inercije segmenata oko centra
mase
L(1).I=[9.12674 8.61229 8.49437 0.16098 -0.57145
1.65126];
L(2).I=[8.19389 0.76659 8.02259 0.42771 0.00337
0.05880];
L(3).I=[6.62158 6.06231 2.08597 -0.10243 -0.02078
-0.02078];
L(4).I=[0.08972 0.09288 0.02856 0.00000 0.01767
0.00000];
L(5).I=[0.03819 0.06668 0.04821 0.00000 -0.00191
0.00000];
L(6).I=[0.00130 0.00131 0.00243 0.00000 0.00000
0.00000];
fanuc=SerialLink(L);
fanuc.name='Fanuc S-430iF';
fanuc.manuf='Fanuc Robotics';
fanuc.plot3d(q);

```

Activating the *Run* command from a palette of tools, the code is executed as defined in scripts. This process creates a structure *Fanuc* robot manipulator, which is a condition that creates a robot model in *Simulink*.

3.2. Creating Simulink model

The next step in creating models is the insertion of the necessary blocks. That can be done by opening the library Robotics Toolbox. It is possible to insert the model in Simulink blocks necessary to define trajectories, calculation of direct and inverse kinematics and dynamics, as well as the simulation for the robot manipulator in a three-dimensional display system. In the library are blocks that allow you monitoring changes in position, velocity, acceleration and moments in time for the individual joints, as well as display trajectory actuating element in a space or in certain sections in Cartesian coordinate system.

The first block that should be included in the Simulink model is *jtraj*. With this block is used to input coordinates (angles of rotation or linear movement,

depending on the type of joint, rotation or translation) for each joint. In this way, define the start and end position of the manipulator, and specifies the time for which is going to be executed move from one given position. This block calculates the generalized coordinates of the joints (in their coordinate system) for all positions between the two set positions: initial (q_0) and end (q_f), resolution for a given displacement, or calculated the trajectory for each joint individually (Fig. 8).

Fig. 8: *jtraj* block parameters

The other necessary blocks have to be added: *fkine* that calculates direct kinematics. This block name has to be changed with the name of object that is created in script, *fanuc*. In order to enable a graphical representation of a robot, it is necessary to import block plot from the library Robotics Toolbox. As with the previous block, this needs to change its name to *fanuc*.

Fig. 9: Block diagram with first three blocks

After setting the connection parameters of the initial three blocks it is possible to achieve a visual representation of the model, by running the command Run, after which the result is display robots as Fig 10.

Block *fkine* solve the direct kinematics problem. For each position of joint defined by block *jtraj* obtained position for manipulator. The direct and inverse kinematics calculation of the robot is obtained by calling the function of a $T=fkine(fanuc,q1)$ and $qi=ikine(fanuc,T)$ in the script. The result calculated from direct kinematics obtained transformation matrix that allows conversion the position end-effector (executive element) from its (moving) coordinate system to the stationary (base) coordinate system, at any moment of time, for a given resolution.

Fig. 10: Robot model figure

Display in a Cartesian coordinate system can be achieved by using block *T2xyz* located in the Robotics Toolbox library. Each position on the trajectory it is possible to be visually monitor in the block *plot*.

If you need a display trajectory end-effector in the projected plane with the three-dimensional system, it is necessary to insert from the *Sinks* library block *XY Graph*. By linking the blocks with the block *T2xyz* is possible to create a movement diagram in one of the Cartesian plane.

If you need to track a numerical values for position, it is possible if you add the block *Display* in *Simulink-Sinks* library.

Speed calculation can be done by differentiating the directions of movement, while the acceleration is calculated by differentiating the velocity joints. The block which performs differentiation *Derivative* is located in *Simulink-Continuous* library.

For a graphical representation of some variables during the simulation is necessary to use blocks from the *Simulink* type *Scope-Sinks* library. Linking them to particular output ports, you can monitor the desired quantities (position, speed, acceleration, torque) in real time.

Inverse dynamics calculation can be achieved with block *rne* from the library Robotics Toolbox. This calculation is obtained by calling the function: `invD=rne(fanuc,q,qd,qdd)` in the script.

Result dynamic calculation, the values torques that must be achieved at each joint in order to make movement along a given trajectory, for every moment in time defined by the given resolution. For this calculation it is necessary to have a defined trajectory, speed and acceleration for each joint as input parameters, while the output gain moments can be displayed graphically using *Scope* block. Final Simulink model of the robot manipulator is shown in Fig 11.

After creating a model in *Simulink* you can simulate movement for the robot manipulator S-430iF.

Fig. 11: Final Simulink block diagram

4. MOVEMENT SIMULATION

To create a better presentation and a more efficient simulation of movement, the new version of Robotics Toolbox (9-10) have improved ability to import 3D models of individual manipulator segments. If you want to make 3D simulation of your robotic manipulator, you have to run script which must have command `.plot3d`. It is a new command in Robotics toolbox that allows the simulation of the robot perform with the imported 3D solid models. Adding a 3D solid model into the simulation gets a complete visual experience of robot. To be able to use this method, it is necessary that all models have names `linkn.stl`, where n is the number of axes (starting from 0 - basics robot to 6 - the last joint). Within this method it is necessary to define the exact path to the model segments and trajectory at which the robot will move. Optional parameters of this method are: colour, desktop size, the distance of the robot from the ground, pause between frames, frame rate, etc. In Fig. 12 shows the Simulink model formed by imported 3D models generated in SolidWork.

Launching files with Simulink model, manipulator S-430iF model is placed in the starting position, defined by block `jtraj` [0 -pi / 2 0 0 0 0]. Before starting the simulation it is necessary to adjust the duration in which simulates the manipulators operation. It is necessary that the time is the same as time specified in block `jtraj`. You can then run the Simulink model by activating the Run command. Running the simulation, the robot executes a movement from initial to final position [0 pi / 2 pi / 4 3 * pi] defined block `jtraj`.

Fig. 12: Fanuc S-430iF 3D model

Figures 13-15 shows the locations of manipulators in certain points in time, during the manipulator movement in defined trajectory. Figures contains the angles of rotation, speed, acceleration and moments of individual joint, as well as display the end-effector trajectory in the XZ and YZ planes in those moments.

Fig. 13: Robot simulation with diagrams that show changes in the angles of rotation, speed, acceleration and moments of individual joint, as well as display the end-effector trajectory in the XZ and YZ planes. $t = 1.6s$

Fig. 14: Robot simulation with diagrams that show changes in the angles of rotation, speed, acceleration and moments of individual joint, as well as display the end-effector trajectory in the XZ and YZ planes. $t = 5s$

Fig. 15: Robot simulation with diagrams that show changes in the angles of rotation, speed, acceleration and moments of individual joint, as well as display the end-effector trajectory in the XZ and YZ planes. $t = 10s$

5. CONCLUSION

The integration of two differently oriented software in the complex mechanisms design process, made it possible to analyze all the necessary elements in order to determine the optimal parameters of the system. The industrial robots, as a typical representative for complex mechanisms with six degrees of movement freedom, carried out the definition, analysis and simulation of the considered mechanism. Virtual model with a control algorithm implemented in the MATLAB, as an open kinematic chain. This software solved the matrix algebra problem and using the add-on Robotics Toolbox conducted a simulation of the behaviour or the model verification which provides analysis for the real robots behaviour. Industrial robot FANUC S-430iF, described in the whole process from solving complex mechanisms kinematics and dynamics problem with the help of computers, to defining the trajectory and transformation matrix, through solving direct and inverse kinematics and dynamics, as well as the movement simulation.

The basic approach in this study is the implementation of virtual models [20-22], which in this case will not only have the task of visually illustrate the segments construction of the mechanism, but also to describe and simulate the physical behaviour, that is great importance for proper dimensioning and defining individual segments system, and observing the functioning of the whole mechanism. Also, unlike physical models, virtual models can take advantage from modern computer technology, which allows the focus in the design process to move from physical to virtual environment. Virtual models can be used for simulation and verification of system before the realization a physical model, thus achieving considerable savings in time and resources required to create and modify physical models.

REFERENCES

- [1] Brezina, T., Jablonski, R.: Recent Advances in Mechatronics, ISBN 978-3-642-05021-3, Springer-Verlag Berlin Heidelberg, 2009.
- [2] Rao, J. S., Dukupati, R.V.: Mechanism and Machine Theory, ISBN 81-224-0426-X, New Age International, Ltd., Publishers, New Delhi, 2006.
- [3] García-Prada, J.C., Castejón, C.: New Trends in Educational Activity in the Field of Mechanism and Machine Theory, ISBN 978-3-319-01835-5, Springer International Publishing, Switzerland, 2014.
- [4] Viadero-Rueda, F., Ceccarelli, M.: New Trends in Mechanism and Machine Science: Theory and Applications in Engineering, ISBN 978-94-007-4901-6, Springer Dordrecht Heidelberg, New York, London, 2013.
- [5] Milićević, I., Savković, S., Šekarić, M., Slavković, R., Dučić, N., Popović, M.: The use of virtual models in the design of mechanisms, The Eighth Triennial International Conference Heavy Machinery HM 2014, ISBN 978-86-82631-74-3, Proceedings, Session E: Mechanical design and mechanics, pp.29-34, Zlatibor, Serbia, 25-28 June, 2014.
- [6] Gilat, A.: MATLAB: An Introduction with Applications (4th edition), ISBN 978-0-470-76785-6, John Wiley & Sons, Inc., 111 River Street, Hoboken, 2011.
- [7] Hasbun, J.E.: Classical Mechanics with MATLAB applications, ISBN 0763746363, Jones and Bartlett Publishers, Sudbury MA, 2009.
- [8] Wilson, H.B., Turcotte, L.H., Halpern, D.: Advanced mathematics and mechanics applications using MATLAB, ISBN 1-58488-262-X, Chapman & Hall/CRC, A CRC Press Company, Boca Raton, Florida, 2003.
- [9] SimMechanics™ Getting Started Guide, TheMathWorks, Inc., USA, 2012.
- [10] Corke, P. I., A Robotics Toolbox for MATLAB, IEEE Robotics and Automation Magazine, Vol.3, No.1, pp.24-32, March 1996.
- [11] Denavit, J., Hartenberg, R. S., A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices, Trans. of ASME J.App. Mech. vol. 77, pp. 215-221, June 1955.
- [12] Lombard, M., SolidWorks 2010 Bible, ISBN: 978-0-470-55481-4, Wiley Publishing, Inc., Indianapolis, Indiana, 2010.
- [13] Lombard, M., SolidWorks Surfacing and Complex Shape Modeling Bible, ISBN: 978-0-470-25823-1, Wiley Publishing, Inc., Indianapolis, Indiana, 2008.
- [14] Pancoast, D., Advanced Part Modeling, Dassault Systèmes SolidWorks Corporation, Concord, Massachusetts, USA, 2009.
- [15] Golubović D., Milićević I., Primena matrica transformacija pri rešavanju kinematike i dinamike manipulacionih robota, Tehnički fakultet, Čačak, 2009.
- [16] Robotic and Automation Handbook, edited by Thomas R. Kurfess, CRC Press, 2005.
- [17] Fu, K. S., Gonzalez, R. C., Lee, C. S. G., Robotics: Control, Sensing, Vision, and Intelligence, Mc Graw-Hill Book Company, ISBN 0-07-100421-1, 1987.
- [18] Legnani, G., Righettini, P., Zappa, B., Casolo, F., A homogeneous matrix approach to 3D kinematics and dynamics, Mechanisms and Machine Theory (the scientific journal of IFTOMM. Pergamon Press U.K.) vol.31, No.5, pp.573-605, 1996.
- [19] Vujičić, V., Proračun kinematike, dinamike i simulacija kretanja višestepenog mehanizma primenom MATLAB-a, diplomski rad, Fakultet tehničkih nauka, Čačak, 2015.
- [20] Mandić, V., Virtuelni inženjering, Mašinski fakultet, Kragujevac, 2007.
- [21] Mandić, V., Erić, D., Adamović, D., Janjić, M., Jurković, Z., Babić, Ž., Ćosić, P.: Concurrent engineering based on virtual manufacturing, Technical Gazette, ISSN 1330-3651, Vol.19, No4, pp.885-892, 2012.
- [22] Milicevic, I., Slavkovic, R., Mandic, V., Jugovic, Z., Popovic, M., Virtual Models Application for the Analysis of Strain-Deformation States in the Process of Metals Processing by Plastic Deformation, IMK-14 – Research & Development in Heavy Machinery 19(2013)3, EN 89-96