MODELLING AND PATH PLANNING OF DELTA PARALLEL ROBOT IN VIRTUAL ENVIRONMENT

Vjekoslav Damic, Maida Cohodar & Avdo Voloder

This Publication has to be referred as: Damic, V[jekoslav]; Cohodar, M[aida] & Voloder, A[vdo] (2018). Modelling and Path Planning of Delta Parallel Robot in Virtual Environment, Proceedings of the 29th DAAAM International Symposium, pp.0149-0156, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-20-4, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/29th.daaam.proceedings.021

Abstract

The paper presents an approach to path planning of industrial robot based on the exchanging the signals between its virtual geometric and kinematic models. The proposed approach is applied to an example of parallel robot ABB IRB 360 in a working environment. It has a parallel structure and is capable of very fast pick and place and assembly operations. The kinematic model has been developed in BondSim environment. The virtual 3D model is created using VTK 3D models of the robot parts by importing *stl* files of the parts from official ABB web site. The 3D model was created in BondSim3D Visual program environment. Validation of the proposed approach was done by simulation. During a simulation runs these two models exchange necessary information.

Keywords: delta parallel robot; kinematic model; visualization; path planning

1. Introduction

The popularity of parallel robots is growing in recent years due to their good performances. The high-speed motion under a high load, good position accuracy and light construction make the parallel robots very good solution in many applications such as laser cutting, pick and place operations, assembly and many others. Hence, they attract attention of many researchers. The well-known Gough-Stewart platform with six degrees of freedom (dof) is analysed in [2] and for which the dynamic model is derived using technique of bond graphs. The most used configuration of the parallel robots are delta robots, which have two platforms, one fixed and joined to a moveable platform by three parallel chains [1], [7], [8], [9], [10], [12], [13], [14] and [15]. Kinematic consideration of the parallel robot is subject of [1], [12]. Kinematic solutions of inverse and forward position revolute-input and prismatic-input delta robot are derived in [1]. Some researchers take into account flexibility of some links [9], [13]. Euler-Lagrange's equation is used to derive dynamic model of robot in [9], where flexibility of the links is modeled by representation of links with beam finite elements. The flexibility of the links is taking into account using a matrix structural technique in [13]. Design of the different control strategy is subject of many papers. Authors of [14] compare fractional order PID and integer order PID controllers for delta robot.

Authors of [4] give model-based control scheme providing better position accuracy compared with position-based control. Analysis of workspace and joint space for different type of parallel robots are presented in [10]. One group of the papers is devoted to the robot design. The original design of the parallel robot which can detect object in workspace is proposed and explained in [8]. Design of non-spherical ball-socket joint is given in [13].

In this paper it is presented approach to define virtual 3D model of delta parallel robot which, in collaboration with Bond Graph model, provides execution of desired task. The approach presented is applied to robot ABB IRB360. The paper is organized as follows. In the next section the kinematic structure of delta robot is explained and the kinematic analysis of the robot has been given. Kinematic model for ABB IRB 360 is developed in form of bond graphs using BondSim [5], [17] from which the joint angles are extracted and communicated to the 3D Visual model. The third section is devoted to 3D virtual modelling of ABB IRB 360 using BondSim3D Visual [5], [17]. For more information on BondSim and BondSim3D Visual the reader can consult web site [17]. Validation of presented methodology is confirmed by numerical simulation in section 4. Final conclusion and recommendations for the future work are given in the last section.

2. Kinematic consideration of delta robot

The delta robot consists of two platforms connected by three symmetric chains, as depicted in Fig.1. The upper platform is fixed; the lower one has three translation degrees of freedom and can move along X , Y and Z axis. Different kinds of joints for delta parallel robots are analysed in the literature. The ball-socket joints with clearance are considered in [13]. In this paper, each of three parallel chains is composed of one upper link connected on the upper side to the fixed platform by a revolute joint and on the lower side to the four-bar mechanism by another revolute joint. The four bar mechanism is connected to the moveable platform also by a revolute joint. Authors in [1] show that, if one of parallel links of four bar mechanism is removed, from kinematic point of view, such configuration will remain the same as the original delta robot. In such case parallel manipulator consists of totally $N_1=15$ revolute joints with 1 dof. According to Kutzbach equation, delta manipulator has

$$\text{number of dof} = 6 \cdot (N - 1) - 5N_1 = 6 \cdot 13 - 5 \cdot 15 = 3, \quad (1)$$

where $N=14$ is number of links including ground. More details about the robot mobility are given in [1].

Fig. 1. Robot manipulator with attached coordinate frames

For kinematical analysis of the delta robot, several coordinate frames are used, Fig.1. The world coordinate frame $O_w X_w Y_w Z_w$ is attached to the fixed platform in such a way that its position and orientation is the same as the fixed frame used in RobotStudio (software for *off-line programming* of industrial robots from ABB) [4]. Three coordinate frames $O_i X_i Y_i Z_i$ ($i=1,2,3$) define the place where three upper arms are attached to the fixed platform:

$$\mathbf{A}_i^w = \begin{bmatrix} \cos(\varphi_i) & -\sin(\varphi_i) & 0 & a \cdot \cos(\varphi_i) \\ \sin(\varphi_i) & \cos(\varphi_i) & 0 & a \cdot \sin(\varphi_i) \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}, \varphi_i = (1-i) \cdot 120^\circ, (i = 1,2,3), \quad (2)$$

where angles φ_i and lengths a and c are denoted in Fig.1.

Each of the coordinate frames $O_i X_i Y_i Z_i$ ($i=1,2,3$) is attached to the upper arm at joint $\mathbf{J}_i$ ($i=1,2,3$) and rotates about Y_i axis by joint angle θ_i ($i=1,2,3$):

$$\mathbf{A}_i^{i'} = \begin{bmatrix} \cos(\theta_i) & 0 & \sin(\theta_i) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_i) & 0 & \cos(\theta_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, (i = 1,2,3). \quad (3)$$

Three coordinate frames $O_i X_i Y_i Z_i$ ($i=4,9,12$) are attached in joints $\mathbf{J}_4$, $\mathbf{J}_9$ and $\mathbf{J}_{12}$ to the four bar mechanism providing the rotation about Y_i -axis:

$$\mathbf{A}_4^{i'} = \begin{bmatrix} \cos(\alpha_1) & 0 & \sin(\alpha_1) & l \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha_1) & 0 & \cos(\alpha_1) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_9^{i'} = \begin{bmatrix} \cos(\alpha_2) & 0 & \sin(\alpha_2) & l \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha_2) & 0 & \cos(\alpha_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_{12}^{i'} = \begin{bmatrix} \cos(\alpha_3) & 0 & \sin(\alpha_3) & l \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha_3) & 0 & \cos(\alpha_3) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

where $\alpha_1=119.18^\circ+J_4$, $\alpha_2=119.18^\circ+J_9$ and $\alpha_3=119.18^\circ+J_{12}$ (where J_4 , J_9 and J_{12} denote the rotation angles which are fed to the joints $\mathbf{J}_4$, $\mathbf{J}_9$ and $\mathbf{J}_{12}$ of the virtual model).

Rotations in joints $\mathbf{J}_5$, $\mathbf{J}_{10}$ and $\mathbf{J}_{13}$ are described by:

$$\mathbf{A}_5^4 = \begin{bmatrix} \cos(\beta_1) & -\sin(\beta_1) & 0 & 0 \\ \sin(\beta_1) & \cos(\beta_1) & 0 & -d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_{10}^9 = \begin{bmatrix} \cos(\beta_2) & -\sin(\beta_2) & 0 & 0 \\ \sin(\beta_2) & \cos(\beta_2) & 0 & -d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_{13}^{12} = \begin{bmatrix} \cos(\beta_3) & -\sin(\beta_3) & 0 & 0 \\ \sin(\beta_3) & \cos(\beta_3) & 0 & -d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (5)$$

where β_i ($i=1,2,3$) are angles of rotations about Z_5 , Z_{10} and Z_{13} -axis, respectively. These angles also describe rotation in joints $\mathbf{J}_8$, $\mathbf{J}_{11}$ and $\mathbf{J}_{14}$, ie. β_1 is rotation about Z_8 -axis, β_2 is rotation about Z_{11} -axis and β_3 is rotation about Z_{14} -axis.

To develop inverse kinematic model which describes the joint angles θ_i ($i=1,2,3$) as function of point's coordinate $\mathbf{P}$ of the moveable platform (X_P , Y_P and Z_P), the following the idea given in [1], the next three vector equations expressed in the world frame can be written as:

$$\mathbf{BK}_i + \mathbf{K}_i \mathbf{P}_i = \mathbf{BP}_i, (i = 1,2,3). \quad (6)$$

Vectors $\mathbf{BK}_i$ are actually the radius vectors of the origins of the frames attached to the joints $\mathbf{J}_4$, $\mathbf{J}_9$ and $\mathbf{J}_{12}$ and their components are the first three elements of the forth column of matrices: $\mathbf{A}_4^w = \mathbf{A}_1^w \cdot \mathbf{A}_1^{i'} \cdot \mathbf{A}_4^{i'}$, $\mathbf{A}_9^w = \mathbf{A}_2^w \cdot \mathbf{A}_2^{i'} \cdot \mathbf{A}_9^{i'}$ and $\mathbf{A}_{12}^w = \mathbf{A}_3^w \cdot \mathbf{A}_3^{i'} \cdot \mathbf{A}_{12}^{i'}$, defined by (2), (3) and (4). For the vectors $\mathbf{K}_i \mathbf{P}_i$ and $\mathbf{BP}_i$ yields:

$$\mathbf{K}_i \mathbf{P}_i = [l_{xi} \quad l_{yi} \quad l_{zi}]^T, \quad \mathbf{BP}_1 = \begin{bmatrix} X_P + b \\ Y_P \\ Z_P \end{bmatrix}, \mathbf{BP}_2 = \begin{bmatrix} X_P - b/2 \\ Y_P - \sqrt{3}b/2 \\ Z_P \end{bmatrix}, \mathbf{BP}_3 = \begin{bmatrix} X_P - b/2 \\ Y_P + \sqrt{3}b/2 \\ Z_P \end{bmatrix} \quad (7)$$

Starting from the expression:

$$\mathbf{K}_i \mathbf{P}_i^T \cdot \mathbf{K}_i \mathbf{P}_i = l^2, (i = 1,2,3), \quad (8)$$

and using (6) and (7) the following three equations can be written [1]:

$$e_i \cos(\theta_i) + f_i \sin(\theta_i) + g_i = 0, (i = 1,2,3), \quad (9)$$

where e_i , f_i and g_i ($i = 1,2,3$) are given by:

$$\begin{aligned}
 e_1 &= 2L(a - b - X_p), f_1 = 2L(Z_p - c), g_1 = X_p^2 + Y_p^2 + Z_p^2 + L^2 - l^2 + (a - b)^2 - 2X_p(a - b) + c(c - 2Z_p) \\
 e_2 &= 2L\left(a - b + \frac{1}{2}X_p + \frac{\sqrt{3}}{2}Y_p\right), f_2 = 2L(Z_p - c), \\
 g_2 &= X_p^2 + Y_p^2 + Z_p^2 + L^2 - l^2 + (a - b)^2 + (X_p + \sqrt{3}Y_p)(a - b) + c(c - 2Z_p), \\
 e_3 &= 2L\left(a - b + \frac{1}{2}X_p - \frac{\sqrt{3}}{2}Y_p\right), f_3 = 2L(Z_p - c), \\
 g_3 &= X_p^2 + Y_p^2 + Z_p^2 + L^2 - l^2 + (a - b)^2 + (X_p - \sqrt{3}Y_p)(a - b) + c(c - 2Z_p).
 \end{aligned} \tag{10}$$

Solution of (9) represents inverse kinematic model:

$$\theta_i = 2 \operatorname{atan} \left(\frac{-f_i - \sqrt{e_i^2 + f_i^2 - g_i^2}}{g_i - e_i} \right), (i = 1, 2, 3). \tag{11}$$

3. 3D Virtual model

The 3D virtual model of the robot ABB IRB 360 is developed in graphical environment of BondSim3D Visual using 3D CAD models of the robot parts [4], Fig.2. Program BondSim3D Visual uses VTK C++ library [16] to represent 3D bodies in computer virtual space. The relative position and form of the bodies in the scene are described by simple text scripts. When the scripts are read by the program the corresponding objects are created in the computer memory and represented on the screen as 3D virtual geometric objects. To move these 3D bodies over the screen we use another model, one that describes the kinematics and dynamics of the bodies in the question. To that purpose we use BondSim program to define the corresponding kinematics of the problem. We also may add the complete dynamics of the problem [5], [17]. These two programs – 3D virtual and Bond Graph – run on the same or different computers linked by a local net and communicate using *named pipes*. During the simulation of the BondSim model, the model regularly collect the values of the joint angles and send them to the 3D virtual model to move the corresponding bodies over the scene. 3D model by itself also returns back information on position of the set probes. The two-way communication between the programs executes typically every 20 ms. The motion of the objects can be also recorded and replayed later.

Fig. 2. Virtual model of ABB IRB 360 workspace

We now show scripts files used to define ABB IRB 360 robot in its working environment. The robot system is defined by several script (txt) files. Table 1 shows script which defines the robot workspace. The first command *Axes* define the *word* coordinates. It is defined by ZYZ Euler transform with respect to default coordinate frame having X-axis directed to right, Y-axis vertically upward, and Z-axis out of the screen. Thus it is necessary to rotate default coordinate frame for -90° about its Z-axis, then for -90° about new Y-axis, and finally for 90° about new Z-axis.

Parameter *lbox* defines extend of the axes. Next three *Insert* instructions define the objects that are put into workspace: the robot ABB IRB 360, and two accessories. The positions of the accessories are specified by shift commands with respect to world coordinates.

Axes (euler -90 -90.0 90.0 lbox 800); Insert robot ABB_IRB360_X_800_STD; Insert accessory Conveyor_belt (shift x -500 y -500 z -1190); Insert accessory Sensor (shift x -500 y -175 z -1150);
--

Table 1. The robot system structure

The virtual 3D model of the robot manipulator ABB IRB 360 was described by a script, a part of which is shown in Table 2. The first line defines *Device* that describes the robot. Basically we define the coordinate frames using *Joint* commands with specified number (usually it means the order of the joints appearance in the structure) and arguments specifying the joint frame position and orientation given by ZYZ Euler angles, with respect to the previously *defined* coordinate frame. The joint type is denoted at the end of the instruction. Robot ABB IRB 360 has three parallel structures. The branching is indicated by *Extend* instruction indicating from which the next joint extends, the base or any other frame specified in the argument of this instruction. The robot parts defined by extern *stl* (stereo-lithography) files and imported into the program are specified by *Part* instruction. The parts can be composed also of elementary shapes such as cubes, cylinders etc.

Device ABB_IRB360_X_800_STD Joint 1 (shift x 200 z -275) revolute y Extend base Joint 2 (shift x -100 y -173.2 z -275 euler 0 0 -120) revolute y Extend base Joint 3 (shift x -100 y 173.2 z -275 euler 0 0 120) revolute y Extend 1 Joint 4 (shift x 235 z 0 euler -180 -119.18 -180) revolute y Joint 5 (shift y -50 z 0) revolute z Joint 6 (shift x 800) revolute z Joint 7 (shift y 50 euler 0 -119.18 0) revolute y Extend 4 Joint 8 (shift y 50) revolute z Extend 2 Joint 9 (shift x 235 z 0 euler -180 -119.18 -180) revolute y Joint 10 (shift y -50 z 0) revolute z Extend 9 Joint 11 (shift y 50) revolute z Extend 3 Joint 12 (shift x 235 z 0 euler -180 -119.18 -180) revolute y Joint 13 (shift y -50 z 0) revolute z Extend 12 Joint 14 (shift y 50) revolute z Extend base Joint 15 (shift x -300 y -420 z -1150) prismatic y Part IRB360_1_800_STD_3D_BASE_02; Part IRB360_1_800_STD_3D_UA1_02; Part IRB360_1_800_STD_3D_LA1_1_02; Set IRB360_1_800_STD_3D_LA1 add IRB360_1_800_STD_3D_LA1_1_02; Part IRB360_1_800_STD_3D_PLATE_02; cylinder endeffector dia 30 res 28 len 47.07; Set IRB360_1_800_STD_3D_PLATE add IRB360_1_800_STD_3D_PLATE_02 endeffector (shift z -1013.54);
--

```

Part IRB360_1_800_STD_3D_LA1_2_02;
Part IRB360_1_800_STD_3D_UA2_02;
.....
cube WP 100 150 50;
Set Workpiece add WP;
.....
Set ABB_IRB360_X_800_STD add IRB360_1_800_STD_3D_BASE_02 (shift z 0 euler 90 -90 -90);
Render ABB_IRB360_X_800_STD
    color 0.8 0.8 0.8;
.....

```

Table 2. The robot system structure

Development of the robot structure is finished by setting the parts into the appropriate robot frames (*Set* 'name of robot' *add* 'name of part') and rendering them in the scene using *Render* instructions and specifying the colour.

4. Numerical example

To verify proposed approach a simple task is defined. The end effector approaches to the workpiece during the first 3 s of the simulation, which is denoted by red dash line in Fig.2. End-effector starts its motion at point 1 passes through point 2 and arrives at point 3 at the end of the third second. At the same time, when the end-effector begins its motion, the workpiece starts to move at a velocity of 175 mm/s by the conveyor-belt. The proximity sensor is mounted on the conveyor (marked with blue in Fig.2). Sensor detects the presence of workpiece and produces a signal to stop the workpiece's motion. The parameters for robot ABB IRB 360 (in Fig.1) are defined as $a=200$ mm, $b=45$ mm, $c=-275$ mm, $L=235$ mm, $l=800$ mm and $L_{\text{end-effector}}=41.97$ mm [4].

During simulation the kinematic model waits for sensor's signal to stop workpiece. When signal is delivered from the virtual model to the kinematic one, the kinematic model gives instruction to stop of workpiece motion. On the end of 3rd s of the simulation the end-effector comes in contact to the workpiece and stamp some mark on it (point 3, Fig.2). After that the end-effector returns back (point 4, Fig.2). Trajectory of the end-effector is given by:

$$\begin{aligned}
 & \begin{cases} -150 \cdot t \\ -125 \cdot t \\ -5.9117 \cdot t^3 + 31.3817 \cdot t - 973.5 \end{cases} & 0 \leq t \leq 2 \text{ s} \\
 [X_p] &= \begin{cases} -300 \\ -250 \\ -100 \cdot (t - 2) - 958.03 \end{cases} & 2 < t \leq 3 \text{ s} \\
 [Y_p] &= \begin{cases} -300 \\ -250 \\ 100 \cdot (t - 3) - 1058.03 \end{cases} & 3 < t \leq 4 \text{ s} \\
 [Z_p] &
 \end{aligned} \tag{12}$$

To provide that the end-effector follows this trajectory on the virtual side, kinematic model sends signals with desired values of joint angles using IPC (inter process communication) to the virtual side (IPC component is drawn as a red ring in Fig.3a). Much more about the established communications is given in [3], [5] and [6].

Fig. 3. a) Kinematic model developed by BondSim; b) Coordinates of point P X_p , Y_p and Z_p Time

Coordinates of point **P** obtained during simulation and that send back from the virtual side are compared. The result of comparison (Fig.3b) shows very good agreement confirming validity of proposed approach. Note that there is a time delay between these two signals of about 20 ms.

After simulation has finished ($t=4s$), point **P** has coordinates $[-300 -250 958.03]^T$. The values of joint angles at this moment are: $\theta_1=56.776^\circ$, $\theta_2= -26.656^\circ$ and $\theta_3=31.507^\circ$ that are depicted on numerical displays of kinematical model (developed by BondSim), Fig.3a. These values are compared with joint angles θ_i ($i=1,2,3$) obtained by RobotStudio [4].

Using instruction of RobotStudio, the end-effector is moved to reach the coordinates which corresponded to the coordinates of point **P** at the end of simulation. Coordinates, shown in Fig.4a, are coordinates of tool centre point TCP, because the length of end-effector is taken into account. As point **P** is the centre of lower platform on which end-effector is mounted, point **P** has the same *X*- and *Y*- coordinates as TCP. *Z_P*- coordinate of point **P** can be easily calculated as $Z_P=Z_{TCP}+L_{end-effector} = -999.97+41.97= -958.00$ mm. Values of the joint angles, which correspond to these coordinates of point **P**, are taken from RobotStudio and are presented in Fig.4b.

Fig. 4. a) Coordinate of TCP point; b) Joint angles

Comparison of value of joint angles obtained by BondSim and BondSimVisual and ones by software for off-line programming of ABB industrial robots RobotStudio, confirms that proposed approach can be used with confidence.

5. Conclusion

The paper presents approach for task planning of the robot system, based on collaboration between two robot models: the kinematic and the virtual. Models are developed using two different software packages and run in different windows during simulation. Established communication between them provides exchanging information, necessary to define and perform the task. Of course, the quantities delivered from the virtual model can be calculated on the kinematic side. But, sometimes, data can be easily obtained from the virtual model. Also, we plan to extend research and to use the kinematic model for control the real system where we will collect signals from the real system instead of the virtual model. Virtual model will be used to test the kinematic model before we start work with the real system.

Procedure for creating of the virtual model of robot is applied on the fast, very popular industrial robot with parallel structure ABB IRB 360. We plan to develop the dynamic model of this robot manipulator automatically from geometric, virtual model in the future investigation.

6. References

- [1] www.ohio.edu/people/williar4/html/pdf/DeltaKin.pdf, (2016). R.L. Williams II, "The Delta Parallel Robot: Kinematics Solutions", Internet Publication, January 2016., Accessed on: 2018-06-01
- [2] Damic, V. & Cohodar, M. (2015). Dynamic analysis of Stewart platform by bond graphs, *Procedia Engineering*, Vol.100, pp.226-233. (<http://www.sciencedirect.com/science/article/pii/S1877705815003896>)
- [3] Damic, V. & Cohodar, M. (2017). Multibody System Modeling, Simulation, and 3D Visualization, Chapter 17, pp.627-671, in book © Springer International Publishing Switzerland 2017, W. Borutzky (ed.), *Bond Graphs for Modelling, Control and Fault Diagnosis of Engineering Systems*, DOI 10.1007/978-3-319-47434-2_17
- [4] www.abb.com Accessed on: 2018-04-15
- [5] Damic, V. & Montgomery, V. (2015). *Mechatronics by Bond Graphs: An Object-Oriented Approach to Modelling and Simulation*, Springer, ISBN-10: 3662569671, Berlin Heidelberg
- [6] Damic, V.; Cohodar, M. & Damic, D. (2014). Multibody Systems Dynamical Modeling and Visualization based on IPC technique, The proceedings of the 2014 International Conference on Bond Graph Modeling and Simulation – ICBGM'2014, Edited by Jose J. Granda, Dean C Karnopp, Simulation Series, Vol.46, No.8, pp. 773-778, ISBN: 978-1-63266-700-7, The Society of Modeling & Simulation, Monterey, USA, July 6-10, 2014, http://www.daaam.info/Downloads/Pdfs/proceedings/proceedings_2016/014.pdf
- [7] Poppeova, V.; Uricek, J.; Bulej, V. & Sindler, P. (2011). Delta robots – robots for high speed manipulation, *Technical Gazette* 18, 3, 435-445, ISSN 1330-3651

- [8] Apostolescu, T.C.; Ionascu, G. & Duminica, D. (2017). Design and Programming of a New 4-DOF Parallel Robot, ECAI 2017 - International Conference – 9th Edition, Electronics, Computers and Artificial Intelligence, 29 June - 01 July, 2017, Targoviste, ROMÂNIA, 978-1-5090-6458-8/17
- [9] Kuo, Y-L. (2016). Mathematical modeling and analysis of the Delta robot with flexible links, Computers and Mathematics with Applications 71, 1973–1989,.doi.org/10.1016/j.camwa.2016.03.018
- [10] Jha, R.; Chablat, D.; Baron, L.; Rouillier, F. & Moroz, G. (2018). Workspace, joint space and singularities of a family of delta-like robot, Mechanism and Machine Theory 127, 73–95, doi.org/10.1016/j.mechmachtheory.2018.05.004
- [11] Damic, V.; Cohodar, M. & Tvrtkovic, M. (2016). Inverse Dynamic Analysis of Hobby Robot uArm by Matlab/Simulink, Proceedings of the 27th DAAAM International Symposium, pp.0095-0101, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-08-2, ISSN 1726-9679, Vienna, Austria DOI: 10.2507/27th.daaam.proceedings.014,
- [12] Brinker, J.; Corves, B. Takeda, Y. (2018). Kinematic performance evaluation of high-speed Delta parallel robots based on motion/force transmission indices, Mechanism and Machine Theory 150, 111–125, doi.org/10.1016/j.mechmachtheory.2017.11.029
- [13] Yang, E.C-Y.; Yen, M-H.; Chou, M-T.; Huang, Z-J. & Chen, S-K. (2017). Non-spherical ball-socket joint design for Delta-type robots, Mechatronics 45, 14–24, doi.org/10.1016/j.mechatronics.2017.05.003
- [14] Angel, V. & Viola, J. (2018), Fractional order PID for tracking control of a parallel robotic manipulator type delta, ISA Transactions 2018 Aug;79:172-188., DOI:10.1016/j.isatra.2018.04.010
- [15] Kuo, Y-L. & Huang, P-Y. (2017). Experimental and simulation studies of motion control of a Delta robot using a model-based approach, International Journal of Advanced Robotic Systems November-December 2017: 1–14, DOI: 10.1177/1729881417738738
- [16] The VTK User's Guide, 11th Ed., Kitware, Inc., 2010.
- [17] www.bondsimulation.com, Accessed on: 2018-04-15