

DYNAMIC ANALYSIS AND 3D VISUALIZATION OF MULTIBODY SYSTEMS

Vjekoslav Damic^(a), Maida Cohodar^(b)

^(a) University of Dubrovnik, Cira Carica 4, Dubrovnik, Croatia,

^(b) University of Sarajevo, Faculty of Mechanical Engineering, Vilsonovo setaliste 9, Sarajevo, Bosnia and Herzegovina

^(a) vdamic@unidu.hr, ^(b) cohodar@mef.usa.ba

ABSTRACT

3D Visualisation is a powerful tool for dynamic analysis of multibody systems such as robots. The basic idea of the paper is to develop the models of multibody systems from two points of view: the dynamic and geometric. The dynamic model is based on bond graph technique and is developed using a suitable bond graphs modelling and simulation environment (BondSim). To develop 3D model in a virtual scene another program the BondSimVisual is used. It is based on VTK visualization pipeline technology. These two models interact by two-way inter process communication (IPC) based on the named pipes. Thus, two models appear as a single complex model running on the same or separate computers connected by a LAN. The attention of this paper is focused on development of visual 3D models of the multibody systems. The proposed approach is illustrated on example of a pendulum and industrial robot ABB IRB 1600.

Keywords: Multibody systems, 3D visualization, Bond Graph modelling, Inter process communications

1. INTRODUCTION

With increase of computer power, the processes of modelling and simulation become a powerful tool in dynamic analysis of engineering problems. Often it is not necessary to perform expensive experimental work with real system. The benefits of visualization are numerous. It enables representation of the system in a more familiar form. It can also help to recognize possible collisions between the bodies and thus enable better modelling of the problem.

These benefits are recognized by many software designers and they are included in many software packages for dynamic analysis, e.g. MATLAB/Simulink, Dymola, 20-SIM, etc. In MATLAB/Simulink, the bodies of multibody system can be represented as convex hull from body coordinate system locations or equivalent ellipsoid from mass properties. Also, bodies can be built by external graphical 3D model, developed in appropriate 3D CAD software packages, e.g. Solid Works, Catia etc. During simulations, animation of motion automatically starts allowing insight into system movements. Dynamic

model of PUMA560 with Robotics Toolbox for Matlab is given in (Corke 2011). A new robot modelling and simulation toolbox for Matlab devoted to robot control design is presented in (Dean-Leon Nair and Knoll 2012). KUKA control toolbox for motion control of KUKA robot manipulators is given in (Chinello Scheggi Morbid and Prattichizzo 2010).

The tool for translation of CAD models developed in CATIA to Modelica in order to get information about parts and joints and how they are related, is explained in (Elmqvist Mattsson and Chapuis 2007). The issue in design of a virtual reality robot implemented in Java 3D using a modified subsumption control architecture and the problems that may be appear with 3D platform are described in (Smith Egert Cuddihy and Walters 2006).

One of the first software packages which introduced a simulation based on CAD models and integrating the robotic cells and flexible manufacturing systems was GRASP (Graphical Robot Application Simulation Package), of BYG Systems ltd. The robot manufacturers develop their own software for off-line robot programming: ABB Robot Studio, Fanuc RoboGuide, Kuka KukaSim, Motoman MotoSim EG etc. Overview software and visualization technique applied in robotics is presented in (Zlajpah 2010).

This paper deals with modelling of multibody system from two points of view: the dynamical and geometric. The dynamical models are based on bond graph technique implemented in BondSim program. It supports systematic development of complex dynamic models. Using bond graphs the models are developed using effort/flow paradigm. To obtain the positional information the flows (velocities) are usually integrated. However, it is not easy to generate in this way a complex 3D geometric scene.

Thus, to represent a model of a mechanical system from 3D geometric point of view, a completely different approach is used and is implemented in a program BondSimVisual (developed by the first author). It is based on powerful VTK (Visualization Tool Kit) C++ library (VTK User's Guide 2010). Between these two programs an Inter Process Communication is established, which enables that on one side the dynamical model of the mechanical system drives the

corresponding bodies in a 3D virtual scene and in this way enables visualisation of the system motion.

On the other hand the visual (i.e. geometric) model can return positional information of interest to the dynamic model. Thus, e.g. in case of a collision of two bodies the point of contact and direction of the common normal could be of the interest. Thus, such information can be regularly collected during the simulation and sent back to the dynamic side for further processing. The dynamical process e.g. can restrict the further motion of the colliding bodies in the direction of the common normal and forces the motion only in the common tangential plane, or maybe the bodies reject.

The collision problem is one of problems where both positional and dynamical information are important. It is well known that the computer games typically use the dynamic models for the solution of the collisions as well.

The paper is organized as follows.

Because of its fundamental role we start with Inter Process Communication in Section 2. Next the approach used generate 3D visual models are given in the Section 3. In the Section 4 the method developed is applied to a relatively simple example of a pendulum hitting the ground. The pendulums are often used as the benchmarks for testing the formulations (Damic Cohodar and Damic 2013).

In Section 5, the attention is focused on a real industrial manipulator - ABB IRB 1600. Its dynamic model is developed using bond graphs by BondSim and 3D virtual model by BondSimVisual. Two-way communication is established during simulation between two models. Dynamic and visual model are verified by comparison of the simulation results with the ones obtained using Robot Studio (<http://new.abb.com/products/robotics/robotstudio;> (last approach: May, 09 2015)).

2. CONCEPT OF 3D VISUALIZATION BASED ON IPC

The basic idea is to develop two models of a multibody system – a dynamical and 3D visual, Fig.1. They are defined by two different applications running on the same or distant computers connected by a local net. Between them a two-way inter process communication (IPC) based on named pipe technology is used (Damic Cohodar and Damic 2014).

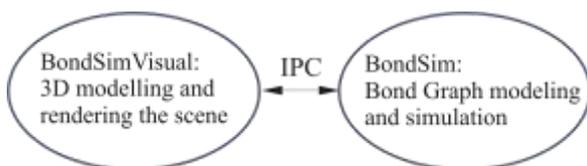


Figure 1: Inter Process Communication (IPC)

The dynamical model using BondSim regularly sends the angular or linear displacements data which are used in the visual space to transform positions of the objects in 3D virtual scene i.e. to rotate and/or translate the

body objects. The visual program responds by redrawing the scene.

On the other hand the 3D model can collect and send back some positional information. In the examples considered the coordinates of a point are picked and returned back. This is of course trivial information, which can be generated in bond graph model easily. It is used here to demonstrate that the both models and IPC work correctly.

The TCP/IP protocols are often used for the IPC. But, for the problems considered it was found that amount of data exchanged is very low for TCP/IP, really often in range of garbage. On other hand it is not permitted to gather data before transmitting it, which is a normal behaviour of TCP/IP; in that case the synchronism between two processes will be lost.

In spite of relatively small amount, the data are exchanged with a great frequency, e.g. every 50 ms or even more frequent. It was found that named pipes are ideally suited for this purpose. Fig. 2 shows the concept of named pipes used.



Figure 2: The configuration of named pipes IPC

The server (BondSimVisual) is responsible for creating the pipe with a specified name. It also creates a special processing thread, which enables that the program simultaneously with other tasks listens for the message from the Client. The server also asks the client (BondSim) to connect. When the client is successfully connected the two-way communications is established. Thus, most of IPC operation is on the server side. But, the client (BondSim) also has a part. To support IPC the client uses an IPC interface component. It serves to define signals which are transmitted to the server by IPC, and also which receive the data from the Server. It is also used to define the name of remote computer where the server (i.e. BondSimVisual) is running. If it is not defined the program assumes that the server is running on the same computer.

3. VISUALIZATION OF MULTIBODY SYSTEMS

Generating a virtual 3D scene in program BondSimVisual is as based on VTK (VTK User's Guide 2010).

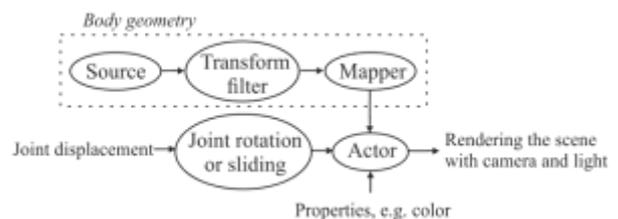
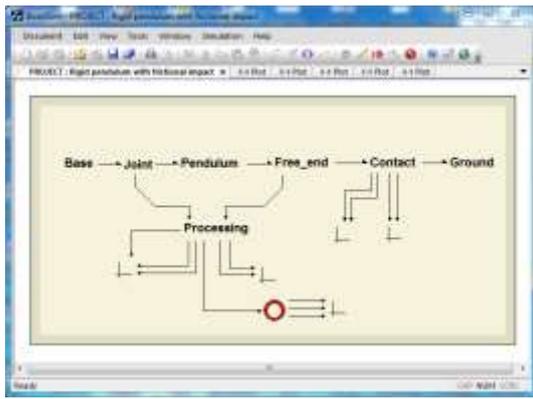
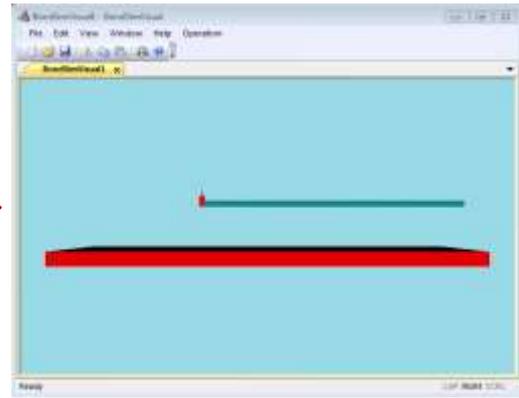


Figure 3: Body visualization pipeline



Dynamic model in bond graph settings, BondSim



3D visual model in BondSimVisual

Figure 4: Visualization of the pendulum

The central structure in VTK is the pipeline of data. The visual representation of a body in a 3D virtual scene with predefined camera and light is shown in Fig. 3. It uses concept of visualization pipelines.

The components in the first row in the figure define the geometry of the body. The *source* enables generation of the body in different forms. It could be primitives such as spheres, cylinders or cubes. But, also more complex forms such as poly-cylinders, or arbitrary modules. CAD models in the form of stl (stereo-lithography) files are supported as well. The *transform filter* is used to transform the body position with respect to a coordinate frame.

The *mapper* is a VTK component that receives data from the filters, or sources directly, and map them into the objects that can be rendered on the screen.

The *actor* is a physical representation of data as they appear in the scene. It accepts data from the mapper, which define the body, but allows also direct interface to the transform that represents the joint rotation or sliding. Also some other properties of the body, such as its colour, can be defined as well. The actors are directly connected to the rendering engine that draws the objects in the scene.

The geometric structure of a multibody system is defined by a script in textual form. It defines the coordinate frames, the bodies it is composed of, their interconnections, colour, etc. Based on this script the visualisation pipeline for the problem is generated and the bodies of the system under the study appear on the scene in its initial position.

4. VISUALIZATION OF A PENDULUM

The proposed approach for creating of visual model is explained on an example of well-known benchmark problem – rigid pendulum with frictional impact. Its dynamic model is developed in (Damic Cohodar and Damic 2014) using the bond graphs (Fig.4 left).

4.1. Bond Graph Model of Pendulum

Initially placed in a horizontal position, the rigid pendulum falls under action of the gravity and hits to the ground. The bond graph model of pendulum is shown in Fig. 4 (left) and consists of three physical components: the Base, Pendulum and Ground. The Base defines the pinned connections of the upper pendulum end. Component Pendulum is developed as a rigid body. Free_End defines the conditions on the body free. The component Contact represents the pendulum friction impact to the Ground.

Mass of pendulum is $m=1$ kg, lengths is $L=1$ m and circular cross area with diameter $D=0.02184$ m. Pendulum hits to the surface at an angle of $\varphi=80^\circ$.

4.2. Development of visual model using basic shapes (model V1)

The visual model can be composed from the basic shapes: cubes, cylinders, poly-prisms, etc. (Fig. 4 right). The coordinates for the pendulum and its components are shown in Fig. 5). They are transformed in the space (translated and rotated) and combined on such way to form the shape of multibody system components. Also, it is possible to create the shapes defined by the modules (defined by the faces, vertices and edges).

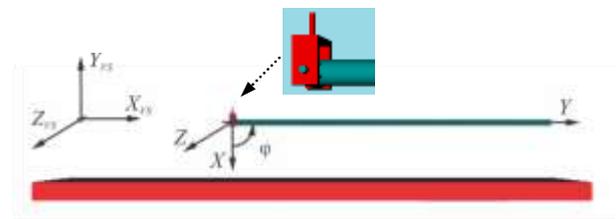


Figure 5: Visual model of the rigid pendulum

Script file of model (denoted V1) that evaluates falling rigid pendulum is given in Table 1. Its dynamic model is developed in coordinate frame whose orientation is shown in Fig. 5 (x -axis is directed down, y -axis to the right and z -axis is out of screen). Its orientation is different from orientation of default virtual screen coordinate frame (the system denoted by index VS in

Fig. 5). Hence, it is necessary to transform the vs frame to our coordinate frame by Euler angles (ZYZ) as given with the last part of the first instruction in the script file (Table 1). Pendulum is connected to the Base by the revolute joint, defined by instruction Joint (with argument in which the joint type and rotation axis are defined). Component Base is composed of three cubes and one cylinder that are set by the instruction “Set Base add...” with the necessary transformations.

Table 1: Script file to define visual model V1

```
!----- Impact_pendulum_V1 -----
Project Impact_Pendulum (euler -90.0 0.0 0.0)
  Joint 1 revolute Z
;
cube T0000170 40.000000 20.000000 4.160000 ;
cube T0000171 40.000000 20.000000 4.160000 ;
cube T0000172 5.000000 20.000000 21.680000 ;
cylinder T0000173 diameter 5.000000 resolution 16 length
20.000000;
set Base add T0000170 (shift x -28 y -10 z 10.84)
          T0000171 (shift x -28 y -10 z -15)
          T0000172 (shift x -28 y -10 z -10.84)
          T0000173 (shift x -28 euler 0 -90.0 0);
cube T0000174 50 1500 500;
Set Ground add T0000174 (shift x173.648 y-500);
Set Impact_Pendulum add Base
          Ground ;
Render Impact_Pendulum
  color 0.91 0 0;
cylinder T0000175 diameter 21.680000 resolution 16
length 1000.000000;
cylinder T0000176 diameter 24.000000 resolution 16
length 21.680000;
cylinder T0000177 diameter 6.000000 resolution 16 length
38.000000;
Set Pendulum add T0000175 (euler -90 -90.0 0)
          T0000176 (shift z-10.84)
          T0000177 (shift z-19);
Set Impact_Pendulum#1 add Pendulum ;
Render Impact_Pendulum#1
  color 0.0 0.6 0.6 ;
Probe Point1 Impact_Pendulum#1(shift y 1000) refer
Impact_Pendulum;
!----- End -----
```

The Ground is set using a simple cube in the appropriate position. The Impact_Pendulum is created consisting of the Base and Ground to which it hits. The model is drawn by instruction “Render”. Its colour is defined by RGB model (red-green-blue values from 0 to 1.0).

Next we define the Pendulum body consisting of three cylinders and added it to the system Impact_Pendulum at the Joint. It is drawn in a different colour.

4.3. Development of visual model using external 3D CAD models (model V2)

We may also develop the geometrical model using 3D CAD models of the Base, Pendulum and Ground, generated in CATIA, as shown in Fig. 6. It could be generated by some other 3D CAD tool (Solid Works,

Pro Engineer, etc.), Models are exported in stl format. In this case the script is very simple as given by table 2. The part names are used to find the stl files in the program the database, which define the parts.

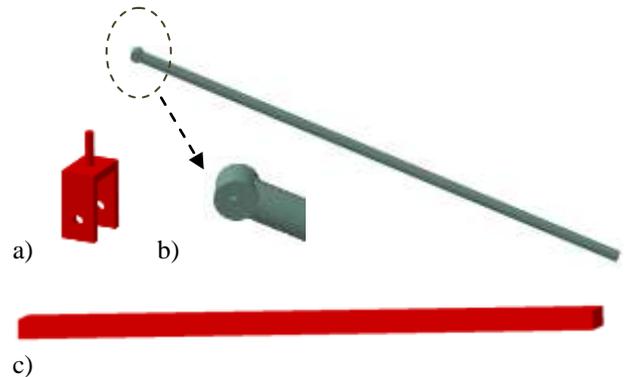
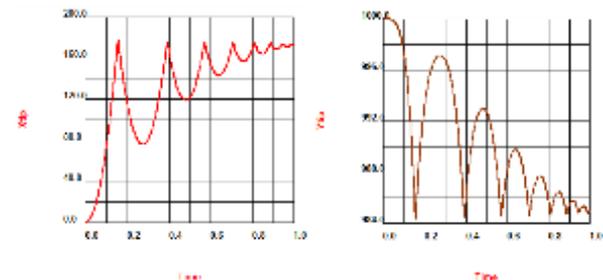


Figure 6: 3D CAD models: a) Base; b) Pendulum; c) Ground

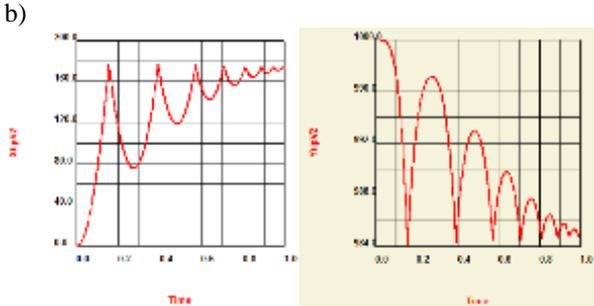
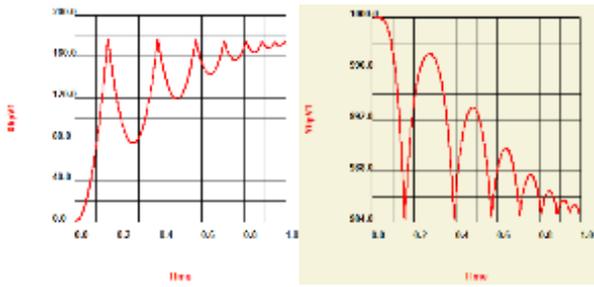
Table 2: Script file for creating virtual model V2

```
!----- Impact_Pendulum_V2 -----
Robot Impact_Pendulum (euler -90.0 0.0 0.0)
  Joint 1 revolute Z
Part Ground;
Part Base;
Part Pendulum;
Set Impact_Pendulum add Base (shift x 12 y -10 euler 0 -
90 0)
          Ground (shift x 173.648 ) ;
Render Impact_Pendulum
  color 0.91 0 0;
Set Impact_Pendulum#1 add Pendulum ;
Render Impact_Pendulum#1
  color 0.0 0.6 0.6 ;
Probe Point1 Impact_Pendulum#1(shift y 1000) refer
Impact_Pendulum;
!----- End -----
```

Two simulations are carried out in order to verify two developed visual models. The last instruction in both script files is “Probe Point1 Impact_Pendulum#1(shift y 1000) refer to Impact_Pendulum;”. With this instruction the coordinates of the pendulum’s tip (the point located at distance of 1000 mm along Y-axis) are picked up from visual model and sent to dynamic side where they are graphically presented on display. In both simulations, the simulation period is 1s, the time step is 1e-5 s, and output interval is 1 ms.



a)



c) Figure 7: Coordinate X_{tip} and Y_{tip} obtained from: a) dynamic model; b) V1; c) V2

Simulation results are presented in Fig. 7. Values of X- and Y-coordinates of tip pendulum obtained on dynamic side are presented in Fig. 7a. Value of X- and Y-coordinates, received from the visual model V1 are shown in Fig. 7b, and for the second visual model V2 in Fig. 7c. The dynamic model of the impact of the pendulum to the horizontal surface is described in detail in (Damic Cohodar and Damic 2013). The comparison of the two approaches for development of the visual model clearly shows that the second one based on 3D CAD models is much more comfortable and simpler to use for the bodies that have complex shapes. But, in the case of relatively simple shapes the first approach is very useful.

5. DYNAMIC AND VISUAL MODEL OF ABB IRB 1600

The robot manipulator ABB IRB 1600 is analysed as the second example. This manipulator is basically composed from seven bodies, Fig. 8. The first body is the base and it is fixed to the ground. The following six bodies are connected to each other by the revolute joints forming an open kinematic chain, as shown in Fig. 9.

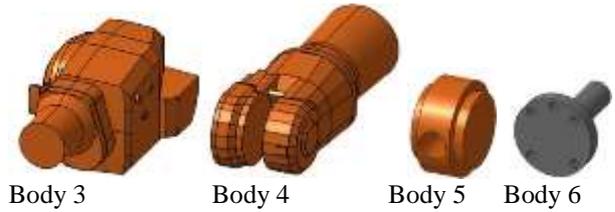
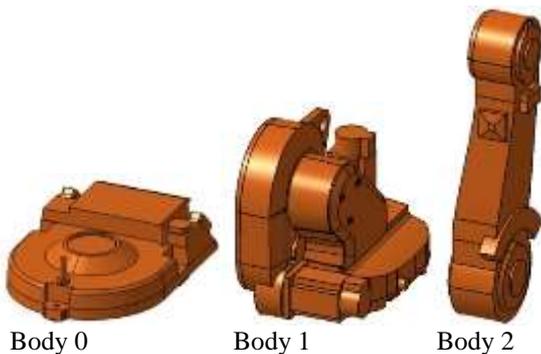


Figure 8: 3D CAD models of robot manipulator ABB IRB 1600

5.1. Direct kinematics of IRB 1600 robot manipulator

The direct kinematic model is evaluated following the Denavit-Hartenberg (DH) formulation and is given in (Cohodar, Begic and Cekic 2014). The coordinate frames are attached to the each body (Fig. 9). The corresponding DH parameters are given in Table 3.

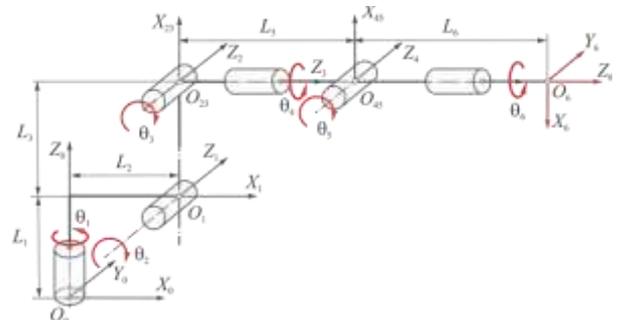


Figure 9: Kinematic scheme of robot manipulator IRB 1600

Table 3: DH parameters for ABB IRB 1600

Link	a_i	α_i	d_i	θ_i
1	L_2	$-\pi/2$	L_1	θ_1
2	L_4	0	0	$\theta_2 - \pi/2$
3	0	$-\pi/2$	0	θ_3
4	0	$\pi/2$	L_5	θ_4
5	0	$-\pi/2$	0	θ_5
6	0	0	L_6	$\theta_6 + \pi$

The matrices of homogenous transformations \mathbf{A}_i^{i-1} ($i=1,2,..6$) are defined for the neighbour frames according to:

$$\mathbf{A}_1^0 = \begin{bmatrix} c_1 & 0 & -s_1 & L_2 c_1 \\ s_1 & 0 & c_1 & L_2 s_1 \\ 0 & -1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_2^1 = \begin{bmatrix} +s_2 & c_2 & 0 & L_4 s_2 \\ -c_2 & s_2 & 0 & -L_4 c_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{A}_3^2 = \begin{bmatrix} c_3 & 0 & -s_3 & 0 \\ s_3 & 0 & c_3 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{A}_4^3 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & L_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

$$\mathbf{A}_5^4 = \begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{A}_6^5 = \begin{bmatrix} -c_6 & s_6 & 0 & 0 \\ -s_6 & -c_6 & 0 & 0 \\ 0 & 0 & 1 & L_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The direct kinematics is obtained by product of matrices

\mathbf{A}_7^{i-1} ($i=1,2,..6$):

$$\mathbf{A}_6^0 = \mathbf{A}_1^0 \cdot \mathbf{A}_2^1 \cdot \mathbf{A}_3^2 \cdot \mathbf{A}_4^3 \cdot \mathbf{A}_5^4 \cdot \mathbf{A}_6^5 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_X \\ r_{21} & r_{22} & r_{23} & p_Y \\ r_{31} & r_{32} & r_{33} & p_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

5.2. Dynamic model of IRB 1600 robot manipulator

Dynamic model of robot manipulator is developed following Newton-Euler formulation in this paper. It is developed by Bond Graphs using BondSim and its system level is shown in Fig. 10 left. Links are

considered as rigid bodies and represented by basic bond graph components of rigid body used from BondSim library. Similarly, joints are presented by bond graph components of revolute joint, also taken from BondSim library. More details about bond graph modelling of a rigid body and joints are given in (Damir and Montgomery 2003).

Unfortunately, we did not have the dynamics data of the robot; but just the geometrical parameters taken from ABB IRB 1600 manual. We know that mass of robot is 250 kg. Hence we imported 3D CAD models of robot bodies (Fig. 8) to CATIA and proportionally to the volumes of the links distribute the masses over the each link. Based on this, we obtained information about masses and moments of inertia.

The interface IPC component (in the form of a ring) sends information about the joint angles to the visual model and receive back X-, Y- and Z- coordinates of the manipulator tip.

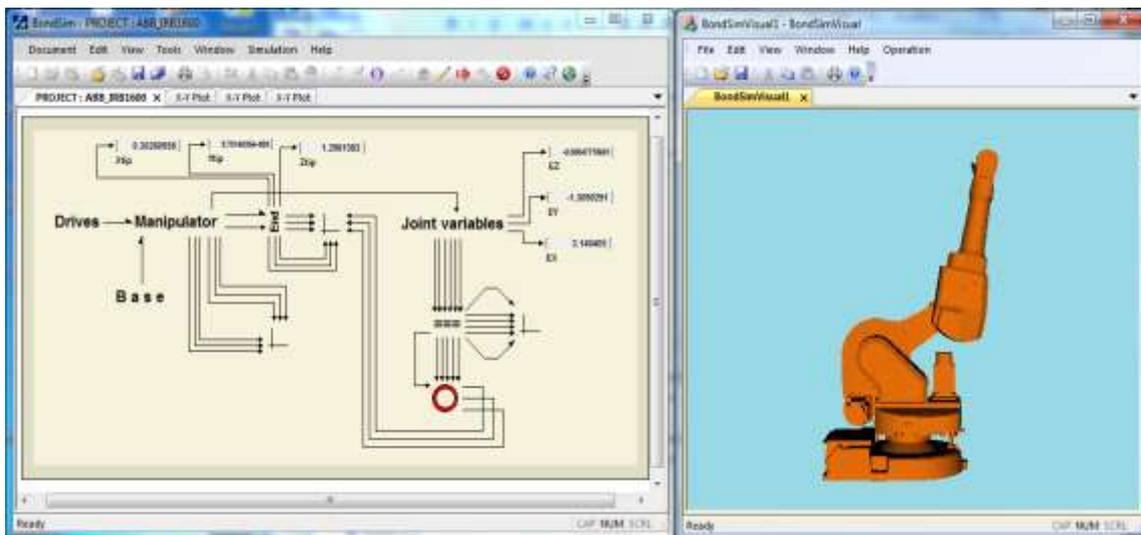


Figure 10: Dynamic and visual models of ABB IRB 1600 robot manipulator

5.3. Visual model of IRB 1600 robot manipulator

As we can see from Fig. 10, the robot links have complex shapes and to develop virtual model of robot we applied second approach from Section 4. 3D CAD models of robot links in stl-format are downloaded from ABB web site:

<http://new.abb.com/products/robotics/industrial-robots/irb-1600/irb-1600-cad>, last approach May, 07 2015 and used in the corresponding script file (Table 4).

Table 4: Script file for creating visual model of ABB IRB 1600

```
!----- ABB IRB1600_12 -----
Robot ABB_IRB1600_12 (euler -90.0 -90.0 90.0)
  Joint 1 revolute Z
  Joint 2 (shift x 150 z 486.5 ) revolute y
  Joint 3 (shift z 475) revolute y
```

```
Joint 4 ( shift x 600 ) revolute x
Joint 5 revolute y
Joint 6 revolute x
initial (0.0 0.0 0.0 0.0 0.0 0.0)
maximum ( 180.0 136 55.0 200.0 115.0 400.0)
;
Part IRB1600_X-120_m2004_rev0_01-1_Body0;
Part IRB1600_X-120_m2004_rev0_01-7_Body1;
Part IRB1600_X-120_m2004_rev0_01-4_Body2;
Part IRB1600_X-120_m2004_rev0_01-5_Body3;
Part IRB1600_X-120_m2004_rev0_01-3_Body4;
Part IRB1600_X-120_m2004_rev0_01-2_Body5;
Part IRB1600_X-120_m2004_rev0_01-6_Body6;
Set ABB_IRB1600_12 add IRB1600_X-
120_m2004_rev0_01-1_Body0 ;
Render ABB_IRB1600_12
color 0.89 0.423 0.039;
Set ABB_IRB1600_12#1 add IRB1600_X-
120_m2004_rev0_01-7_Body1 ;
```

```

Render ABB_IRB1600_12#1
  color 0.0 0.6 0.6 ;
Set ABB_IRB1600_12#2 add IRB1600_X-
120_m2004_rev0_01-4_Body2 (shift x -150 z -486.5) ;
Render ABB_IRB1600_12#2
  color 0.0 0.2 0.8 ;
Set ABB_IRB1600_12#3 add IRB1600_X-
120_m2004_rev0_01-5_Body3 (shift x -150 z -961.5) ;
Render ABB_IRB1600_12#3
  color 0.89 0.423 0.039 ;
Set ABB_IRB1600_12#4 add IRB1600_X-
120_m2004_rev0_01-3_Body4 (shift x -750 z -961.5) ;
Render ABB_IRB1600_12#4
  color 0.1 0.1 1.0 ;
Set ABB_IRB1600_12#5 add IRB1600_X-
120_m2004_rev0_01-2_Body5 (shift x -750 z -961.5) ;
Render ABB_IRB1600_12#5
  color 1.1 1.1 1.1 ;
Set ABB_IRB1600_12#6 add IRB1600_X-
120_m2004_rev0_01-6_Body6 (shift x -750 z -961.5) ;
Render ABB_IRB1600_12#6
  color 0.1 0.1 0.1 ;
Probe Point1 ABB_IRB1600_12#6( shift x 65) refer
ABB_IRB1600_12;
!----- End -----

```

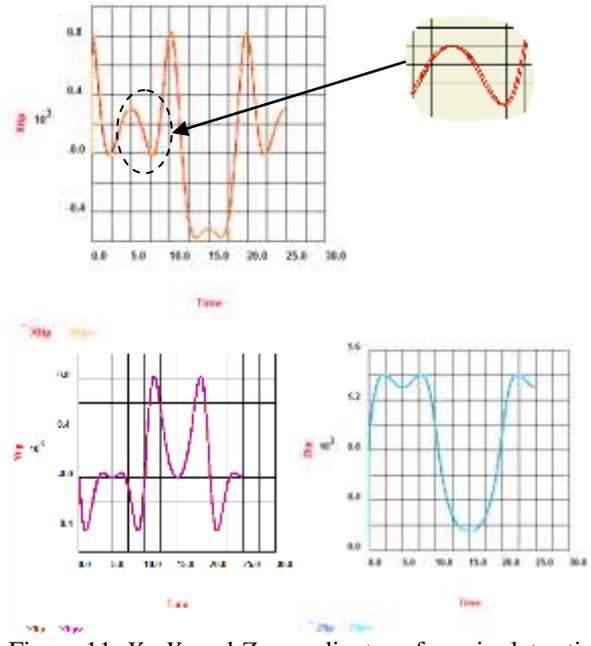


Figure 11: X-, Y- and Z- coordinates of manipulator tip

The simulation interval was set to 25 s. The joints are driven with angular velocities in [rad/s] as follows:

$$\omega_i = -a_i \frac{\pi^2}{10} \cdot \cos\left(\frac{\pi}{10} t\right), (i = 1, 2, \dots, 6), \quad (3)$$

where $a_1=a_4=a_6=1$, $a_2=1/3$, $a_3=2/9$ and $a_5=11.5/18$.

Fig. 11 shows comparison of X-, Y- and Z- coordinates of manipulator tip in the global frame (denoted by index 0 in Fig. 9), which are obtained from the dynamic model and that received from the visual model of the robot manipulator. We can see very good agreement. Also, we can notice that received coordinates from virtual model are shifted in time. This is time required for transfer of necessary information and synchronisation of the motion of geometric and dynamic models.

In order to verify developed dynamic model we compared the values of X-, Y- and Z- coordinates of the manipulator tip from the dynamic model) with that of the exact analytical solution (p_x , p_y and p_z from Eq. 2). Fig. 12a shows that these differences are in order of $1e-6$ m.

Finally, we check our results with the one obtained with RobotStudio (ABB software for off-line robot programming). In workspace of RobotStudio we inserted robot IRB 1600 (1.2), moved the links to set up values of joint angles achieved at the end of the simulation ($\theta_1=\theta_4=\theta_6=-180^\circ$, $\theta_2=-60^\circ$, $\theta_3=-40^\circ$ and $\theta_5=-115^\circ$) and read obtained values of X-, Y- and Z-coordinates and three Euler angles from Virtual Flex Pendant (Fig.12b). These values, read from Virtual Flex Pendant of Robot Studio are in the agreement with the values obtained by the dynamic model (values shown on display components in Fig. 10 left).

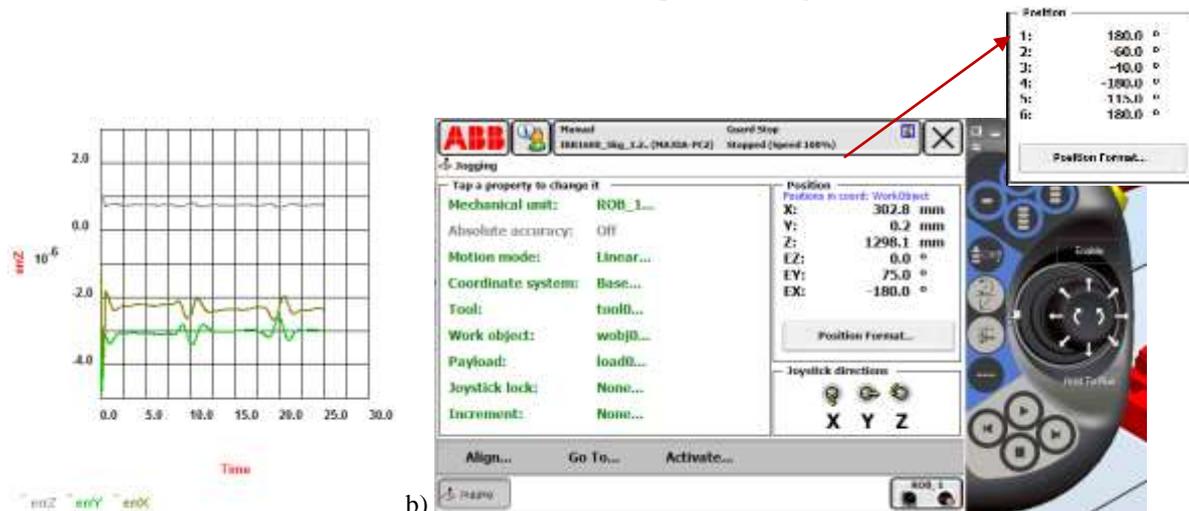


Figure 12: a) Deviation of obtained coordinates by simulations from the analytical solution; b) Position and orientation of end-effector

6. CONCLUSION

Basic idea of the paper is to create two models – dynamic and visual ones using two separate software packages: BondSim and BondSimVisual. Between them there is two-way communications, based on named pipe technology. The paper describes two approaches to developing the virtual models.

One is based on the use of basic shapes, as well as creating some special shapes. Second way uses 3D CAD model developed in appropriate 3D CAD models as Solid Works, Catia etc, that are exported from these software packages in form of stl-file.

Script file is more a simpler in the second case. Proposed approaches are applied to an example of falling pendulum and the motion of industrial manipulator ABB IRB 1600.

ACKNOWLEDGMENTS

This paper is realized in framework of project supported by Federal ministry of education of Bosnia and Herzegovina.

REFERENCES

- Chinello F, Scheggi S, Morbidi F, Prattichizzo D, 2010. The KUKA Control Toolbox: Motion control of KUKA robot manipulators with MATLAB. Robotics and Automation Magazine (ICRA). IEEE International Conference, pp.4603:4608, May 3-7.
- Cohodar M, Begic D, Cekic A, 2014. Machining systems and industrial robots. Sarajevo: Faculty of Mechanical Engineering.
- Corke P, 2011. Robotics, Vision and Control, fundamentals algorithms in Matlab. Springer.
- Damic V, Montgomery J, 2003. Mechatronics by Bond Graphs: An Object-Oriented Approach to Modelling. Berlin-Heidelberg: Springer-Verlag.
- Damic V, Cohodar M, Damic D, 2013. Bond Graph Formulation of Impact with Friction in Multibody Systems, Proceedings of the IASTED (MIC 2013), Acta Press, 794-085, pp. 298:302, February 11-15, Innsbruck, Austria.
- Damic V, Cohodar, M, Damic D, 2014. Multibody Systems Dynamical Modeling and Visualization based on IPC technique, The proceedings of the 2014 Int. Con. on Bond Graph Modeling and Simulation – ICBGM'2014, Simulation Series, The Society of Modeling & Simulation, Vol.46, No.8, pp. 773:778, July 6-10, Monterey, USA.
- Dean-Leon E, Nair S, Knoll A, 2012. User Friendly Matlab-Toolbox for Symbolic Robot Dynamic Modeling used for Control Design. Proceedings of the 2012 IEEE International Conference on Robotics and Biomimetics, December 11-14, Guangzhou, China.
- Elmqvist H, Mattsson S.E, Chapuis C, 2007. Redundancies in Multibody Systems and Automatic Coupling of Catia and Modelica, Proceeding 7th Modelica Conference, pp. 551:560, Sep. 20-22, Como, Italy.
- Karnopp D. C, Margolis D. L, Rosenberg R. C, 2006. System Dynamics: Modeling and Simulation of Mechatronic Systems (4th ed.). Hoboken, NJ: John Wiley & Sons.
- Smith N, Egert C, Cuddihy E, Walters D, 2006. Implementing Virtual Robots in Java3D Using a Sdsumption Architecture, Proceedings from the Association for the Advancement of Computing in Education.
- VTK User's Guide, Install, Use and Extend The Visualization Toolkit, Kitware, Inc., 11th ed., 2010.
- Zlajpah, L, 2010. Robot Simulation for Control Design, Robot Manipulators Trends and Development, Agustin Jimenez and Basil M Al Hadithi (Ed.), ISBN: 978-953-307-073-5, InTech, DOI: 10.5772/9206. Available from: <http://www.intechopen.com/books/robot-manipulators-trends-and-development/robot-simulation-for-control-design>
<http://new.abb.com/products/robotics/industrial-robots/irb-1600/irb-1600-cad>
[http://new.abb.com/products/robotics/robotstudio:](http://new.abb.com/products/robotics/robotstudio;) (approach date: May, 09 2015)