Modelica Model for the youBot Manipulator

Rhama Dwiputra Alexey Zakharov Roustiam Chakirov Erwin Prassler
Bonn-Rhein-Sieg University of Applied Sciences, Department of Computer Science
Grantham-Allee 20, 53757 Sankt Augustin

Abstract

This paper presents the development of Modelica model for the youBot manipulator. Whereas other robotic simulations focus on the robot interaction with its environment, Modelica allows the modeling of the manipulator controllers and motors. The model was developed with a Modelica library for the manipulator's components which provides modularity, reusability and abstraction. A comparison test with the actual system has been performed to ensure the model accuracy. The test shows promising result and provides possible future work. The Modelica model of the youBot manipulator is freely available.

Keywords: Control; Manipulator; Modelica; youBot

1 Introduction

Models and simulation tools are crucial in robotic research. Although there have been major improvements in the electronic and mechanical field, robots are still expensive equipments. The use of models and simulation tools overcome this problem. Models and simulation tools allow researchers and university students to experiment with different robots. Furthermore, experimentation with models is cost-efficient and time-efficient due to its ability to be automated, conditioned and accelerated.

The *youBot* is a mobile manipulator designed to serve as the reference platform for industry, research and education [1]. Due to its frequent use as a test subject for educational purpose or investigation of new methods in research institute, a model for the youBot is highly advantageous. Robotic simulation tools with the youBot model are VREP [3], We-bots [4] and Gazebo [5]. Like most robotic simulation software, these software focus on simulating the robot interaction with its surrounding environment (navigation, object manipulation) and have its limitation when simulating the robot's internal components (mechanical, electrical, and control system). Modeling the robot's internal component requires multi-domain capability such as provided by the Modelica[1] description language. Modelica is a non-proprietary, object-oriented, description language for multi domain modeling. Modelica is maintained by the non-profit Modelica association. As such, Modelica is suitable for use in education and research (similar to the purpose of the youBot

The youBot standard configuration consists of an omnidirectional mobile platform and a five DOFs manipulator with a two finger gripper. In this paper, the development of the Modelica model for the youBot manipulator is presented. The manipulator model is developed by dividing the system into several smaller components. The components models are stored in a new Modelica library and categorized in different packages based on its functionality. This approach enables the user to experiment with the model on the component level.

A model is a representation of the actual system and the benefit of having a model only holds true when the model is accurate. Simulation can result in wrong conclusion when the researcher forget the limitations and condition under which the simulation is valid [2]. Therefore, the development of the manipulator model is followed by a test with the actual system. The test compares the behaviors of the actual system and the model throughout a point-to-point motion. The model accuracy along with the influence of estimated values and approximation is analyzed in the comparison test.

This paper is constructed as follows. After this introductory section, Modelica related robotic research is presented in Section 2. Section 3 presents the specification of the youBot manipulator and Section 4 describes the Modelica Library for the youBot manipulator. Afterward, section 5 presents the evaluation of the developed model. Finally, section 6 summarizes the work and provides possible future work.

[1] www.modelica.org

2 State of The Art

Modelica has been used for modeling spider robotic arm [6], 6-axis industrial robots [7, 8, 9], 3 DOFs parallel Gantry-Tau robot [10], 5 DOFs manipulator [11] and mobile platforms [12, 13]. In most cases, a robot model in Modelica is used for investigating the manipulator's motion control especially in the domain of optimization and system dynamics. Such research requires the repetition of motions and adjustments to the controller which can have damaging effect when being executed on a real robot.

[8] performed optimization through iteration to find a compromise between acceleration, velocity and energy consumption and [9] solved the minimum time optimization problem for an industrial robot. [7] derives the inverse dynamic model of a manipulator using algorithms for differential-algebraic equation available in the Dymola[1] software. Dymola was also used in [11] to design a picking manipulator for agriculture purposes. [10] develops method for kinematic calibration with the Modelica model of parallel Gantry-Tau robot. Aside in the field of motion control, Modelica robot models have also been used for tele-manipulation [6], robot communication [13] and teaching tools [12].

As shown from the work presented in this section, there is a wide range of research with robot models in Modelica. The Modelica model of the youBot manipulator will enable such research to be performed. Since the youBot is designed to be the reference platform for academic institute, the Modelica model of the youBot manipulator is of high importance.

3 The youBot Manipulator

The specification of the manipulator is acquired from the following sources:

- official youBot website[2],
- email communication with the official distributor of the youBot[3] and
- discussion with researchers from BRICS[4] who were involved in the development of the youBot's software.

[1] www.3ds.com/products-services/catia/portfolio/dymola
[2] http://youbot-store.com/
[3] info@locomotec.com
[4] http://www.best-of-robotics.org/

This section consist of two subsections, *kinematic chain* and *control system*. Due to the nature of the robot which is actively being developed, the description presented is subject to changes.

3.1 Kinematic chain

The youBot manipulator is a serial chain manipulator with five revolute joints (shown in Figure 1). The manipulator is equipped with a two-finger gripper as its end-effector and each finger weights 0.001 kg. The fingers' body, position and motion has insignificant influence to the system dynamic when compared to the overall manipulator system. Therefore, the gripper is modeled only for the visualization purpose.

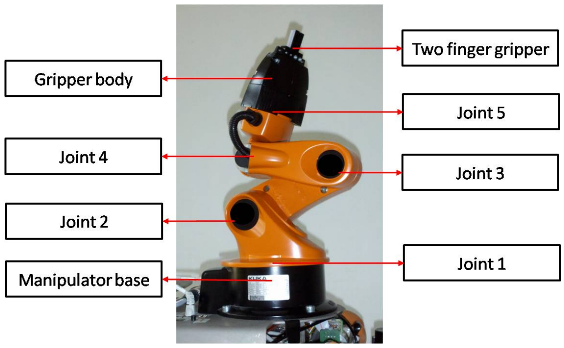


Figure 1: The youBot manipulator

The manipulator is 65.5 cm high when fully extended, weights 6.3 kg and has a payload of 0.5 kg. Each joint is actuated by brushless DC motors and gearboxes with different specifications. The kinematic chain, joint ranges and dynamic properties of the manipulator are presented in appendix A.

3.2 Control System

The control system accommodates position, velocity and current control in each joint. For each joint, the control system consists of: 1. three cascaded *proportional-integral-derivative* or PID controllers, 2. a velocity ramp or v-ramp generator and 3. a space vector pulse width modulation (SVPWM). Two modes are available for joint position control, *PID* and *v-ramp* mode. The PID mode calculates the joint velocity in a PID controller whereas in the v-ramp mode, a trapezoidal velocity profile will be generated by the v-ramp generator for the joint velocity. In this paper, the developed model is based on the joint position control in PID mode. Figure 2 shows the overview of the manipulator's controller.

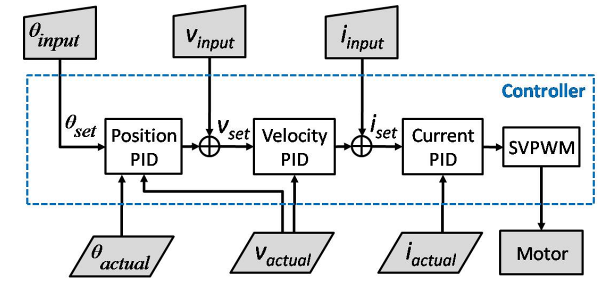


Figure 2: Controller overview

Where θ is the joint angle, v is the joint velocity and i is the motor current. The *set* variables (θ_{set}, v_{set}, i_{set}) are the input values for the PID, the *actual* variables are the values from the manipulator's sensors and the *input* variables are the user defined values. When controlling the joint position, a user provides the θ_{input} for the controller and the *Velocity PID* receive the output of the *Position PID* as its v_{set}. When controlling the joint velocity, a user provide the v_{input} for the controller which is directly forwarded as v_{set} to the *Velocity PID* (the output of the *Position PID* in such cases will be ignored). The *Position PID* is replaced with the v-ramp generator in v-ramp mode. The PID controllers for position, velocity and current have similar architecture. As a representative of the PID controllers, Figure 3 shows the overview of the PID controller for velocity (*Velocity PID*).

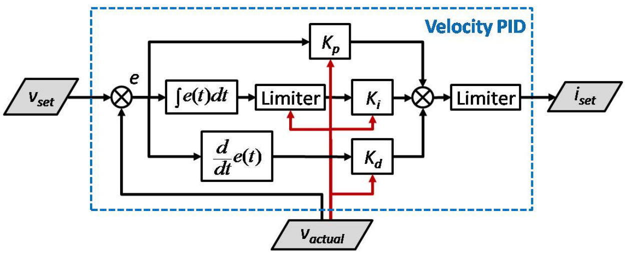


Figure 3: Velocity PID overview

Where e is the difference between the set value and the actual value. K_p, K_i, and K_d are the gain parameters for the controllers. The output of the *Velocity PID* is forwarded to the *Current PID* as i_{set}. As observed in Figure 3, the Velocity PID controller is similar to the text book PID as follows:

$$C_t = K_p e(t) + K_i \int_{t-\Delta t}^{t} e(t)dt + K_d \frac{d}{dt} e(t) \qquad (1)$$

Where C is the controller output and Δt is the PID period. However, the gain parameter in the velocity PID adjusts itself based on the motor velocity as follows:

$$k = \begin{cases} k_2 & \text{if } |v| \geq a \\ k_1 + (\frac{|v|}{a} * (k_2 - k_1)) & \text{if } |v| < a \end{cases} \qquad (2)$$

Where k is the gain parameters (K_p, K_i or K_d in Equation 1), k_1 is the lower boundary of the gain parameter, and k_2 is the upper boundary of the gain parameter value, v is the motor velocity and a is the threshold value. The *Position PID* has the same characteristic as the *Velocity PID*. Therefore, the *Position PID* and the *Velocity PID* are referred as the non-linear PID. The non-linear PID enables the user to set different control behaviors for low and high velocity. Similar to the gain parameters, limiters in the position and velocity controller have non-linear characteristic where the limit value is defined by the motor velocity.

4 The *youBot* Modelica Library

The Modelica library for the youBot manipulator in this paper is developed with Dymola. The library is developed using a "divide and conquer" principle with emphasize on modularity, re-usability and abstraction. This approach enables components exchange and component-based experiment. Additionally, a template model is provided for components which are frequently used in the manipulator model. In such cases, the model has adjustable parameter sets to be configured based on its implementation. Finally, the manipulator model is developed in different abstraction layers (Figure 4). The lower layer provides a more detailed information in each component and the upper layer provides the general overview of the system.

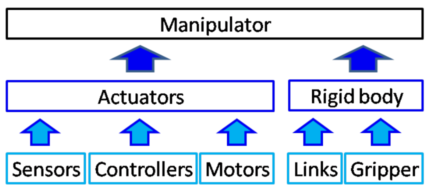


Figure 4: Abstraction layer in a manipulator model

In every modeling process, using estimated values and approximation is unavoidable mainly due to the following reasons:

- *Limited knowledge*. Many parameter set of a dynamic system are estimated through system identification (friction, inertia tensor).
- *Restricted information*. Many manufactures do not provide complete information about their product.

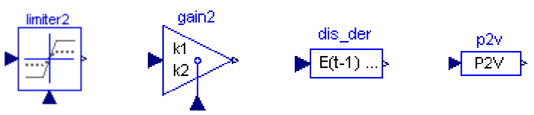
The use of estimated values and approximation is presented in the description of each package. The youBot Modelica library consists of four packages which are:

- *Controller* package,
- *Axis* package,
- *Body* package and
- *System* package.

The library is developed with the use of several packages in MSL such as Modelica.Blocks.Math for standard mathematical functions and Modelica.Mechanics for 3-dimensional mechanical systems. This paper follows the Modelica convention in describing the models. Model's name or package's name begins with capital letter. When necessary, the model includes its package name. The model *Modelica.Blocks.Interfaces.RealInput* refers to the model *RealInput* which is inside the package *Interfaces*. The Interfaces package is inside the *Blocks* package and the *Blocks* package is inside the Modelica Standard Library or MSL. An instance of a model is written in lower case aside from a few exceptional cases (e.g. V is used for voltage to differentiate from v for velocity).

4.1 *Controller* Package

The *Controller* package consists of the components for the manipulator control system. The *Controller* package is divided into three packages which are the *Components* package, the *PIDs* package and the *Modes* package. The *Controller.Components* package consists of models which are in the lowest level of abstraction layer. Figure 5 show the models in the *Controller.Component* package.



(a) *Limiter2* (b) *Gain2* (c) *DisDer* (d) *P2V*

Figure 5: The *Controller.Component* models

Following the Modelica convention, the instance's name of a model is placed on the upper part of the symbols in blue color. The model *Limiter2* and *Gain2* (Figure 5b and 5a) perform the calculation for nonlinear PID controller (Equation 2). The model *DisDer* (Figure 5c) produces the derivative value of a specific time period from a discretized continuous input. The model *P2V* (Figure 5d) converts PWM rate to voltage rate. The *P2V* model is an approximation of the SVPWM component in the controller.

The *Controller.PIDs* package consists of three different PID models which are the *Position*, *Velocity* and the *Current* model. As the name suggests, the models are the PID controllers for position, velocity and current in the youBot manipulator (Figure 2). As a representative, Figure 6 shows the *PIDs.Velocity* model.

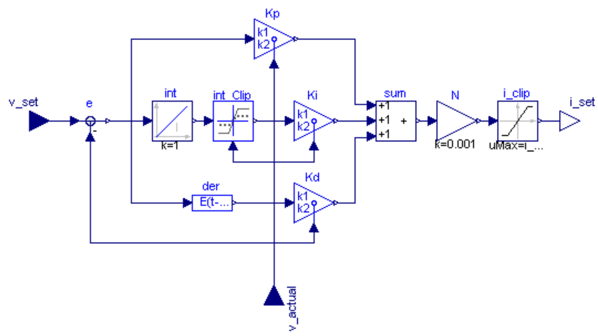


Figure 6: *PIDs.Velocity*

Where *v_set*, *v_actual* and *i_set* represent v_{set}, v_{actual} and i_{set} in Figure 3 respectively. The additional component *N* in the model produces the output in mA to mimic the readings of the actual system.

Finally, the *Controller.Modes* package is for different types of control mode. Currently, the available model in the *Modes* package is the *Position* model. Figure 7 shows the *Modes.Position* model.

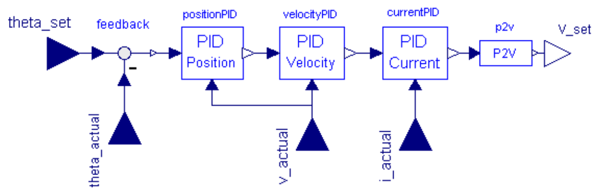


Figure 7: *Modes.Position*

The *Modes.Position* model consist of all three PID models from the *Controller.PIDs* package. *V_set* represent the voltage value which will be connected to the motor's power supply unit. Using the same approach, the model for controlling the joint velocity or the motor current can be developed.

4.2 *Axis* Package

The *Axis* package consists of the model for joint actuator (motor and control system). The package is named *Axis* because the model will be connected to the rotating axis of the manipulator's joints. The *Axis* package consists of the *Position* model shown in Figure 8.

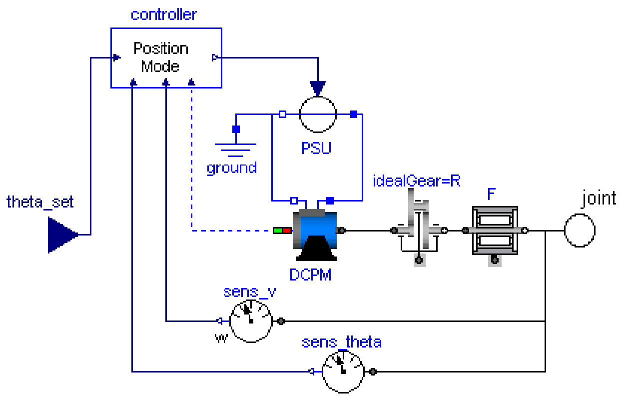


Figure 8: *Axis.Position*

Where *DCPM* represents the brushless DC motor model, *PSU* represents the power supply unit model, *R* represents the gearbox model, *F* represents the friction model, *controller* is the *Modes.Position* model (Figure 7), *sens_v* represent the joint's velocity sensor, *sens_theta* represent the joint's position sensor and *joint* is the connector to the manipulator's joint model. The *controller* output (*V_set* in Figure 7) is connected to *PSU* and its input is extended for the model input as *theta_set*. The output of *sens_v*, *sens_theta* and the value of *DCPM*'s current is connected to the control system (*theta_actual*, *v_actual* and *i_actual* in Figure 7). The model for DC motor, power supply unit and friction uses the ideal system model from the MSL.

4.3 *Body* package

The *Body* package consists of models for the rigid body model of the manipulator's kinematic chain. The *Body* package has three models which are *Gripper*, *Link* and *Manipulator*. The *Body.Gripper* model is the rigid body model of the youBot two finger gripper. The *Body.Link* model is the rigid body model of a manipulator link. Figure 9 shows the *Body.Link* model.

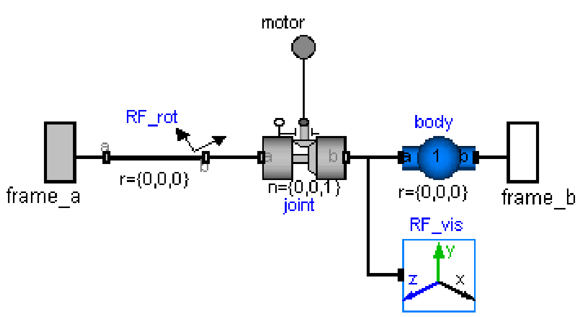


Figure 9: *Body.Link*

Where *RF_rot* is the link's reference frame rotation, *frame_a* is the connector to the previous link model, *frame_b* is the connector to the next link model, *joint* represent the link's joint which is connected to the actuator model through the connector *motor*, and the *body* represent the rigid body of the link. The additional component *RF_vis* provides user with the visualization of the link's reference frame.

The *Body.Manipulator* model represent the rigid body model of the youBot manipulator's kinematic link. Figure 10 shows the *Body.Manipulator* model.

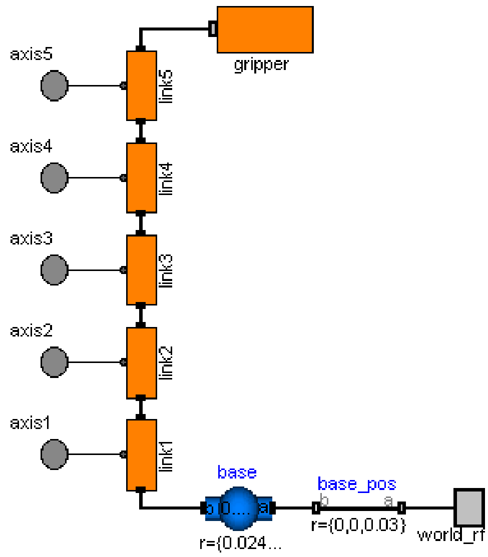


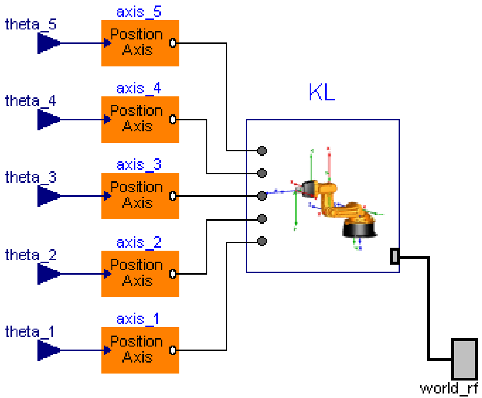
Figure 10: *Body.Manipulator*

Where *link*1 represent the first link of the manipulator (*Body.Link*, Figure 9), *gripper* is the manipulator's gripper model (*Body.Gripper*), *base* represent the rigid body of the manipulator's base. The component *base_pos* is for defining the manipulator position in the world reference frame. The *Body.Manipulator* model has five connectors (*axis*1 to *axis*5) for each joint model and one connector (*world_rf*) for the world reference frame.

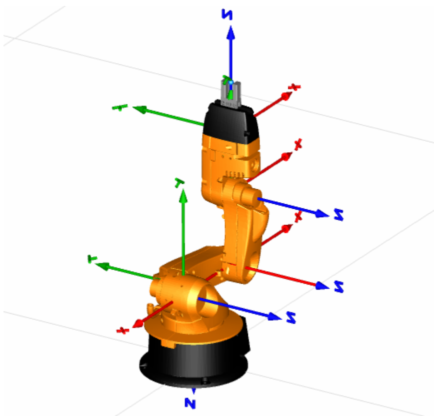
4.4 *System* package

The *System* package consists of the manipulator ready-to-use models. The *System* package has two models which are the *Dummy* model and the *Position* model. The *System.Dummy* model is the rigid body model of the youBot manipulator (*Body.Manipulator*, Figure 10) connected to dummy actuators (*Modelica.Mechanics.Rotational.Speed*). The user can set the velocity of each joint directly in the *System.Dummy* model. The *System.Dummy* model is used for comparison test in Section 5.

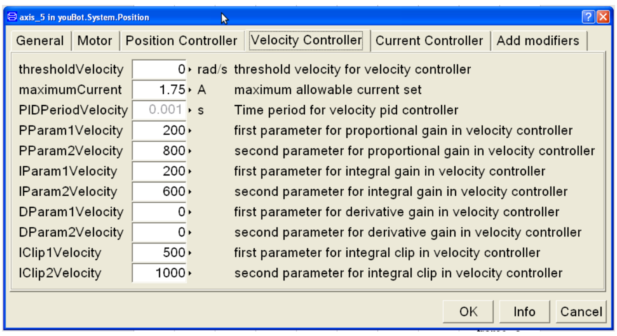
The *System.Position* model is the rigid body model of youBot manipulator's kinematic chain and its actuators. Figure 11 shows the *System.Position* model, its visualization in Dymola and the parameter set configuration for the velocity controller in the manipulator's fifth joint.



(a) *System.Position*



(b) Model visualization



(c) Parameters configuration

Figure 11: The youBot manipulator model

In Figure 11a, *KL* represents the rigid body model of the youBot manipulator's kinematic chain, *theta_5* represents the user defined joint angle and *axis_5* represents the actuator (Figure 8) for joint 5. The parameter names in Figure 11c are consistent with the existing driver and firmware.

5 Comparison Test

A comparison test with the actual system is performed after the development of the manipulator model. The test purpose is to evaluate the model accuracy and identify the major components which require further development. The test involves the comparison of the joint position and the joint velocity throughout a point-to-point motion. For the actual system, the joint velocity is recorded while performing the motion. The sensor measurement of the joint velocity is assumed to be accurate. Afterward, the recorded joint velocity is used as input for the System.Dummy model. In the same setting, the manipulator model (*System.Position*, Figure 11a) is also performing the same motion. The manipulator's joints in this test are set to be frictionless. The motion involves all joints moving 90°. Such motion was chosen so that the resulting error will be the accumulation of the estimated value and approximation in all joints. Figure 12 shows the end-effector paths during the motion.

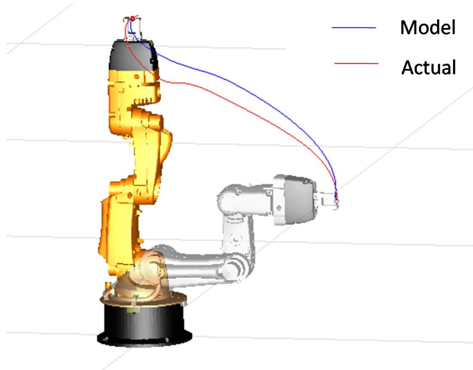


Figure 12: Test result

The gray-colored youBot manipulator is the starting pose of the motion. As expected, the path generated by the model is smoother than that of the actual system as a result of the idealistic conditions in the simulation. Figure 13 shows the error in joint position.

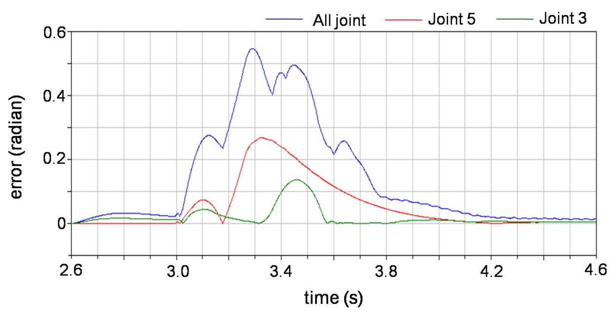
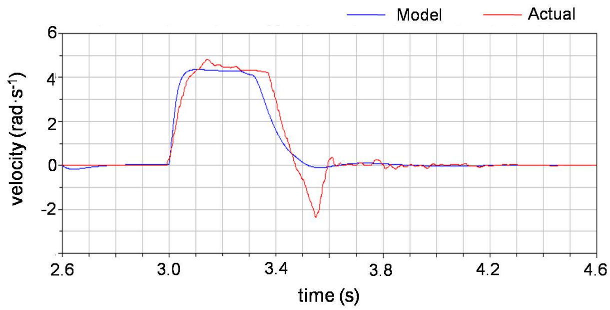
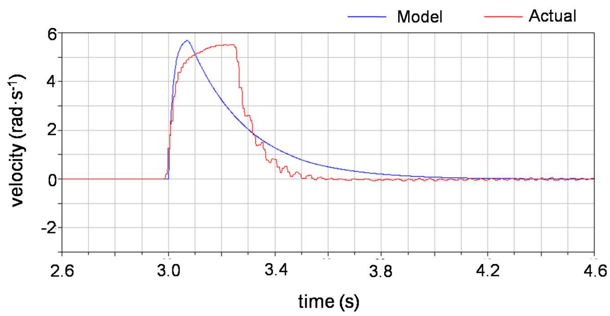


Figure 13: Joint angle difference

The sum of error from all joint peaked at the value of 0.55 radian. The error in each joint depends on the maximum velocity parameter (v_{max}) in the controller. As shown in Figure 13, joint 3 ($v_{max} = 4.19$ rad $\cdot$ s^{-1}) has a considerably lower peak than joint 5 ($v_{max} = 5.90$ rad $\cdot$ s^{-1}). The error in joint angle peaked at two points. Both peak points happened slightly after the velocity change (from stop to move and slowing down from a constant velocity). This is consistent with the error in joint velocity as shown in Figure 14.



(a) Joint 3



(b) Joint 5

Figure 14: Joint velocity comparison

The joint velocity in the actual system is less stable than in the simulation (Figure 14a). This is the result of the motor vibration which is excluded from the manipulator model. The ideal motor model results in the deviation on higher velocity (Figure 14b) which correspond to the higher error in joint position for joints with higher v_{max} value in its controller. Similar phenomena in joint velocity and joint position are also found in other joints. From this result, it is concluded that the motor model is the major factor of the deviation. Other possible contributing aspects in the deviation between the model and the actual system are inertia tensor estimation, SVPWM approximation and frictionless joints.

6 Conclusion

In this paper, the development of Modelica model for the youBot manipulator is presented. The Modelica library for the manipulator components provides the user with modularity, reusability and abstraction. The model accuracy has been evaluated through a comparison test with the actual system. The test result shows that the model reflects the actual system within a reasonable deviation. The analysis shows that the motor model is the major contributing aspect in the deviation between the model and the actual system. Possible improvements for the developed Modelica library is the development of a more accurate motor model and a more comprehensive test for the manipulator components (controller components and dynamic properties of the rigid body model).

For future work, the manipulator model is planned to be used for hardware-in-the-loop experiments. The development or design of other manipulator models is also possible through the reusability of the components model in the Modelica library. The library is publicly available[1] to be used as a teaching tool or for experiments involving manipulator dynamics, load identification, fault analysis and motion control.

References

[1] R. Bischoff, U. Huggenberger, and E. Prassler, "Kuka youbot - a mobile manipulator for research and education," in *IEEE Int. Conf. on Robotics and Automat. (ICRA)*, pp. 1 –4, may 2011.

[2] P. Fritzson, *Principles of Object-Oriented Modeling and Simulation With Modelica 2.1*. IEEE Press, 2004.

[3] M. Freese, S. Singh, F. Ozaki, and N. Matsuhira, "Virtual robot experimentation platform v-rep: a versatile 3d robot simulator," in *Proc. of the Second Int. Conf. on Simul., Model., and Program. for Auton. Robots*, SIMPAR'10, (Berlin, Heidelberg), pp. 51–62, Springer-Verlag, 2010.

[4] O. Michel, "Webots: Professional mobile robot simulation," *J. of Adv. Robotics Syst.*, vol. 1, no. 1, pp. 39–42, 2004.

[5] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-

[1] www.youbot-store.com

robot simulator," in *IEEE/RSJ Int. Conf on Intell Robots and Syst.*, pp. 2149–2154, 2004.

[6] G. Ferretti, M. Gritti, G. Magnani, and P. Rocco, "A remote user interface to modelica robot models," in *Proc. of the 3rd Int. Modelica Conf.*, pp. 231–240, 2003.

[7] M. Thuemmel, M. Otter, and J. Bals, "Control of robots with elastic joints based on automatic generation of inverse dynamics models," in *Proc. of 2001 IEEE/RSJ Int. Conf. on Intell. Robots and Syst*, pp. 925–930, 2001.

[8] A. Kazi, Merk, M. G. Otter, and H. Fan, "Design optimisation of industrial robots using the modelica multi-physics modeling language," in *Proc. 33rd ISR Int. Symp. Robot*, pp. 347 –352, 2002.

[9] M. Hast, J. Åkesson, and A. Robertsson, "Optimal robot control using modelica and optimica," in *Proc. of the 7th Int. Modelica Conf.*, Modelica Association, Sept. 2009.

[10] I. Dressler, J. Schiffer, and A. Robertsson, "Modeling and control of a parallel robot using modelica," in *Proc. 7th Int. Modelica Conf.*, (Como, Italy), Sep 2009.

[11] Y. Chen, S. Jin, X. Zou, D. Xu, and W. Cai, "Study of modeling and simulating for picking manipulator based on modelica," in *Proc. of the 2nd Int. Conf. on Intell. Robotics and Appl.*, (Berlin, Heidelberg), pp. 1211–1216, Springer-Verlag, 2009.

[12] J. Åkesson, U. Nordstroem, and H. Elmqvist, "Dymola and modelica_embeddedsystems in teaching - experiences from a project course," in *Proc. of the 7th Modelica Conf.*, 2009.

[13] U. Pohlmann, S. Dziwok, J. Suck, B. Wolf, C. Loh, and M. Tichy, "A modelica library for real-time coordination modeling," in *Proc. of the 9th Int. Modelica Conf.*, 2012.

A Manipulator Specification

Table 1: Kinematic chain

	Parent frame	Translation (*cm*)			Rotation (degree)		
		x	y	z	x	y	z
Joint 1	Base	2.4	0	11.5	180°	0°	0°
Joint 2	Joint 1	3.3	0	0	90°	0°	-90°
Joint 3	Joint 2	15.5	0	0	0°	0°	-90°
Joint 4	Joint 3	0	13.5	0	0°	0°	0°
Joint 5	Joint 4	0	11.36	0	-90°	0°	0°
Gripper	Joint 5	0	0	5.716	90°	0°	180°

Table 2: Joint range

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5
Joint range	-169° 169°	-65° 90°	-151° 146°	-102.5° 102.5°	-165° 165°

Table 3: Dynamic Properties

	Mass (*kg*)	Intertia Tensor Elements ($kg{\cdot}cm^2$)		
		I_{xx}	I_{yy}	I_{zz}
Link 1	1.39	29.525	60.091	58.821
Link 2	1.318	31.145	5.483	31.631
Link 3	0.821	17.2767	4.1967	18.468
Link 4	0.769	6.764	10.573	6.61
Link 5	0.678	1.934	1.602	0.689
Gripper	0.201	2.324	3.629	2.067