OO Modelling and Control of a Laboratory Crane for the Purpose of Control Education

Borut Zupančič Primož Vintar

Faculty of Electrical Engineering, University of Ljubljana, Slovenia, borut.zupancic@fe.uni-lj.si

Abstract

The paper deals with modelling, simulation and control of a laboratory crane for the purpose of control education. There were many similar activities in the past with realisations in causal modelling e.g. Matlab-Simulink. However we wanted to model and control the set-up also in the OO and multi-domain environment Dymola-Modelica using library components instead of mathematical equations to show all the advantages of such approach. The combination with some causal structures to solve certain problems is also discussed. The model was properly validated with some open and closed loop experiments. These results confirm the applicability of the model and the efficiency of the mentioned approach in modern control engineering courses.

Keywords: control education, modelling and simulation, OO approach, multi-domain approach, model validation

1 Introduction

It is extremely important to use miniature laboratory plants in control education. As such plants are very expensive they usually cover only a part of exercises, the basic part is realised in modelling and simulation environment. There are usually two important areas which have to be covered with education: modelling of real plants and control. Control schemes can be validated in simulation environment or on real plants.

However there are two approaches in modelling and simulation: traditional causal or block oriented or input output modelling which originates in analog simulation, in CSSL standard and is nowadays mostly covered with Matlab-Simulink environment (Moller, 2004; Simulink, 2014). However, more advanced approach is based on acausal, object oriented modelling which represents a multi-domain approach (connection of components from different fields) and giving a possibility of building libraries with reusable components. The most powerful tools are based on Modelica language (Modelica Association, 2010; Fritzson, 2004; OpenModelica, 2012), which is supported with several modelling packages. In our activities Dymola was used (Dymola, 2015). Many experiences with such tools in industrial projects and in education (Zupančič and Sodja, 2013) show, that it is possible to produce complex model in shorter periods while the models are very illustrative as they retain physical structure. However the executable models become very complex, it is usually difficult or impossible detect and solve numerical problems (Sodja, 2012). The practice shows that students are much more motivated when modelling with OO tools. This was evident when our traditional Matlab-Simulink modelling courses were expanded with Dymola-Modelica several years ago.

In our courses AMIRA 600 laboratory set-up (Amira, 2001) for basic and also more advanced courses from modelling and control was used. It enables several subprocesses, one is so called Loading bridge which actually models a crane. The mathematical model and the implementation in SIMULINK is described in (Hančič et al., 2015).

This paper describes our efforts to model the mentioned set-up without numerous mathematical equations. Instead a modeller uses prepared components from standard Modelica library. Our goal was only partly fulfilled because some modelling problems were finally solved with causal components.

2 Description of the loading bridge

AMIRA 600 (Amira, 2001) (see Figure 1) is a convenient laboratory set-up, appropriate for basic and also more advanced courses from modelling and control. It consists of an aluminium frame covered with sheets of plexiglass. The plant has different configurations, one of them is also the bridge crane. The set-up is very convenient for modelling and control. There are different control tasks such as proper positioning of the load in minimal time, small angles of the thread, avoiding obstacles, etc.

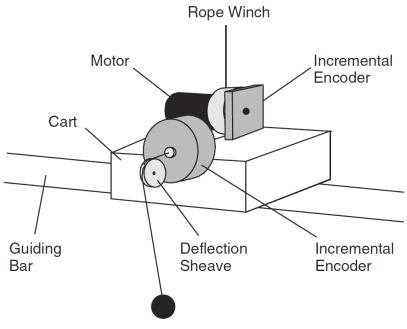

Figure 1. Laboratory set-up for control education: loading bridge.

The laboratory set-up consists of a cart which can be moved along a metal guiding bar by means of a transmission belt. Two proximity switches are mounted close to both ends of the guiding bar. They are used for limiting the position of the cart. The cart carries a rope winch that is used to change the length of the rope. A weight is fixed to another end of the thread. So the length of the thread influences the position of the weight. The lifting and descending of the weight as well as the movement of the cart is realised with two current DC motors enabling the proper positioning of the load (weight). So the DC voltages applied to these motors represent two inputs while the torques produced by both DC motors are proportional to the DC voltages. The outputs are given by three incremental encoders which measure the position of the cart, the length of the thread and the angle between the thread direction and vertical direction (φ). The thread itself together with the load is denoted as a pendulum system. The origin of the coordinate system is the cart in the very left position and the point in which the pendulum is handled to the cart. Figure 2 illustrates the described set-up with some dimensions.

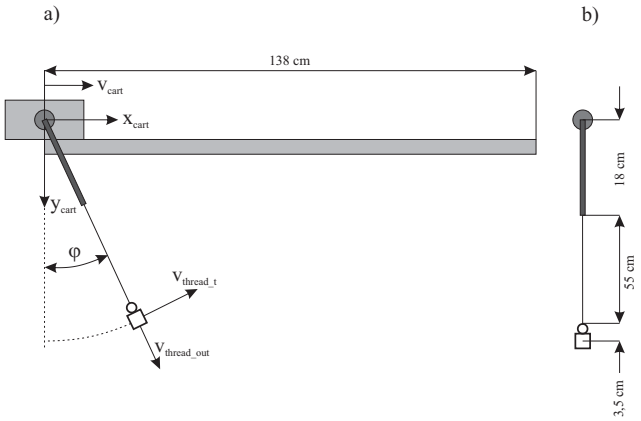

Figure 2. Part a: set-up Amira 600 working as the loading bridge. Part b: pendulum with appropriate dimensions.

Table 1. Physical parameters of the set-up.

Parameters	Values
Mass of the cart+winch	5,7 kg
Mass of the load (weight)	0,143 kg
Length of the guiding bar	1,38 m
Length of the pendulum stick	0,18 m
Distance between the centre of load gravity and the point where the load is fixed to the thread	0,035 m
Maximal length of the rope	0,55 m

Table 1 shows some important plant parameters, which were partly measured or obtained from the documentation (Amira, 2001).

3 Modelling with Modelica

The basic aim of this investigation was to show that Modelica is a convenient approach especially in the modelling of mechanical systems. We intended to use the Modelica standard library and to build an efficient model without usual mathematical modelling with equations. Of course we have to be aware that also a profound knowledge of the mathematical modelling using balance equations in modelling subjects is needed. Even very sophisticated libraries are usually not sufficient the introduction of some changes or developments of customized components in real applications is often needed.

The approach was to build the model with diagram layer (icon or graphical based modelling) and to establish some more efficient solutions using some improvements with textual model layer. Basically the components from two libraries (packages): `Mechanics` and `Blocks` were used.

3.1 Package of mechanical components `Mechanics`

The package `Mechanics` is a part of Standard Modelica library. The library consists of three sub libraries: `Multibody`, `Rotational`, `Translational`.

The library `MultiBody` is a free Modelica package providing 3-dimensional mechanical components to model in a convenient way mechanical systems, such as robots, mechanisms, vehicles. The components - the coordinate system of the world frame, the revolute joints and the rigid bodies have also animation properties.

The libraries `Translational` and `Rotational` are also free Modelica packages which enable modelling of traditional rotational and translational problems which are commonly needed especially in basic modelling courses. The basic components are mass, inertia, spring, damper etc.

It is convenient to model 3D mechanical systems with components from all three libraries, as there are property defined connectors which enable proper conections. As an example we can use two components from the sub library `Mechanics.MultiBody.Joints`, shown in Figure 3. Both components entitled `prismatic` and `revolute` are important in the loading bridge model.

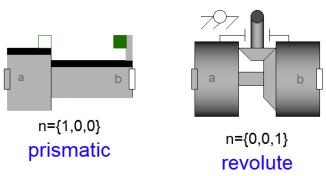

Figure 3. Two components from the sub library `Modelica.Mechanics.Joints`.

The prismatic joint has 1 translational degree-of-freedom. There are two regular connectors (frame `a` and

frame b) which enable to connect the components from the library Multibody. Optionally, two additional 1-dimensional mechanical connections can be driven with elements of the Translational library. This is especially convenient to connect a control force to the 3D mechanical system.

Revolute joint has 1 rotational degree-of-freedom where frame b rotates around axis n which is fixed in frame a. Optionally, two additional 1-dimensional mechanical connections can be driven with a component from Rotational library. This is often used to connect a control torque to 3D mechanical systems.

3.2 Package of causal components Blocks

The library Blocks contains input/output blocks to build up block diagrams. This is actually the implementation of a traditional modelling approach based in causal input/output interactions originating from analog simulation, supported later by a CSSL standard and finally implemented also in Simulink environment. The library is important for the implementation of model parts where the causality is needed, e.g. for the implementation of a control system where inputs and outputs have to be strictly defined.

With components from the library Blocks some new components were implemented and some existing were modified. These components were included into a new library AdditionalComponents, which is a part of the loading bridge model.

3.3 Structure and components of the overall model

Figure 4 shows the hierarchical structure of the loading bridge model. It is implemented with the package CraneModel on the highest hierarchical level. It consists of three sub-packages: AdditionalComponents is the library of components, developed or updated from the existing components from the Standard Modelica library. The second package PilotPlantAMIRA600 is intended to final Amira 600 models. Currently it contains only the model of the loading bridge. The third subpackage Controller is intended to implemented controllers.

3.4 Model of the loading bridge

Figure 5 depicts the top level model of the loading bridge, implemented with model class LoadingBridgeCrane. All components except CartDrive and ThreadDrive are taken from the standard Modelica library Mechanics and model the whole mechanical system with drives. The components CartDrive and ThreadDrive were placed to the library of new and appropriately modified components (AdditionalComponents).

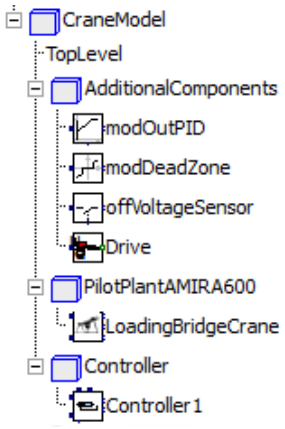

Figure 4. Hierarchical structure of the loading bridge model.

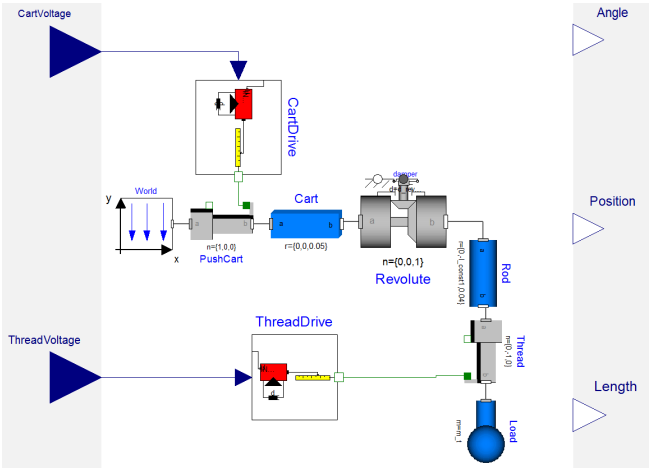

Figure 5. Model of the loading bridge in Dymola-Modelica.

3.4.1 World

Model class World is a part of the standard library Modelica.Mechanics.Multibody. It represents a global coordinate system and the gravity field.

3.4.2 Cart

The cart (model class Cart) is realised with BodyBox component from the Multibody library. The parameters are dimensions, mass, animation features etc. The cart is driven with PushCart element, which is the appropriate actuator for the movement of the cart. The information for the needed movement is calculated in the model class CartDrive.

3.4.3 Pendulum

On the connector b of the Cart the component Revolute is connected. This is a revolute joint which handles the pendulum. The pendulum is built according to the construction shown in Figure 2, b. It is a mathematical pendulum with the whole mass concentrated in a point at the end. The component Rod is a rod with no mass, with appropriate length and it handles the thread which is

realised with the class `Thread`. This prismatic actuator is the implementation of the winch. The movement of the the thread is therefore implemented in the same way as the movement of the cart. The component `ThreadDrive` is almost identical to the component `CartDrive`. So the length of the thread depends on the input DC voltage. The load of the crane is realised with the element `Load` with the whole mass concentrated in the center of gravity point. The distance between the mass point (center of gravity) and the point where the mass is connected to the thread is illustrated in Figure 2, b.

3.4.4 Drive

The model class `Drive` is used in two almost identical classes `CartDrive` and `ThreadDrive`. It transforms the input DC voltage signals into appropriate positions. It consists of causal blocks. The Modelica diagram is shown in Figure 6. This component includes many parameters,

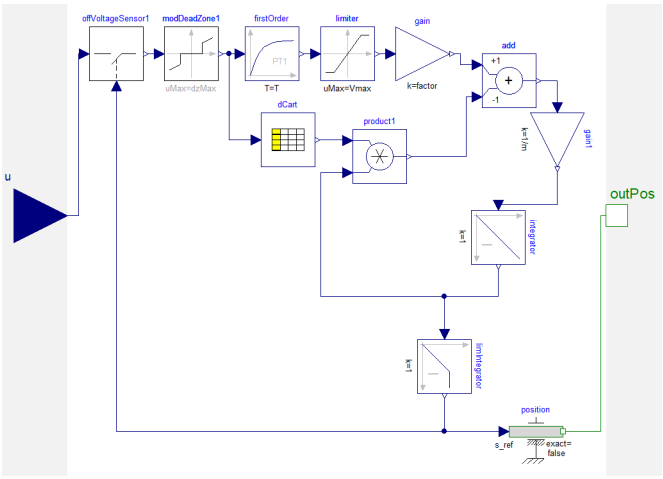

Figure 6. The diagram of model class `Drive` in the Dymola-Modelica environment.

which are needed for proper transformation. The information of the mass of the object and its initial position are also needed. The damping is also important. The experiments show that constant damping is not appropriate. Better results were achieved with the damping which depends on the input voltage. Experimentally obtained curves ware realised with look up tables.

The input to the model is a real variable, representing the DC voltage to the DC motor. This real variable feeds the model class `offVoltageSensor` which models proximity final position switches for limiting the final positions. The output of this class is connected to the input of the component `modDeadZone`, which is a modified standard block `deadZone` from the library `Modelica.Blocks.Nonlinear`. A deadzone in which the voltage is not sufficient for the movement is implemented with this model. The output of the block `modDeadZone` is connected to the lag system `firstOrder` with an appropriate time constant. With this time constant (delay) it was possible to tune the speed of the positioning of the cart and the length of the thread. At the beginning we tried to compensate a fast dynamics with the appropriate damping but it appeared that the appropriate delay, which has to be included, does not depend on the velocity. The output from the lag system goes to the limiter of the input voltage (`limiter`) and is finally in the block `gain` transformed into the appropriate force. The gains 5.17 for the cart and -0.35 for the thread were obtained experimentally. This gains actually represent very simplified models of DC motors. Additionally we improved the model by introducing a damping force. If a force F acts to a mass m, which movement is also damped (damping b), then the acceleration can be evaluated with the equation

$$F - bv = ma \tag{1}$$

where v denotes the velocity and a acceleration. However the model was improved with nonlinear damping where b depends on the DC voltage. The nonlinear function was realised with the look up table `dCart`. Finally the acceleration was obtained by dividing the left hand side of Eq. 1 with the mass m. With two integrations the velocity and the position are obtained (model classes `integrator` and `integrator1`). The calculated position was connected to the position component (`position`) which interacts the causal and acausal modelling parts and implements appropriate movement of the cart or thread.

It is clear that the described modelling is not what was intended at the beginning - to implement fully acausal model, because it introduces partly causal modelling and reduces the efficiency of Modelica language. However this was the most efficient solution of the problem appearing above all in conjunction with the pendulum. Namely we were not able to compensate gravitation and centrifugal forces of the pendulum in open loop experiments what resulted in uncontrolled and unstable movement of the thread. With the solution that the driving actuators obtain the information of the position instead of the force, the compensation forces are automatically generated inside actuators.

4 Controller in Matlab Simulink environment

Dymola-Modelica is an extremely powerful tool for true physical modelling. However for complex experimentations (e.g. optimisation, linearisation, steady state calculation, etc.), for results presentation, e.t.c. it is far from Matlab possibilities. So we decided to use Dymola-Modelica just for the 'physical' part and Matlab-Simulink for all other needs: Simulink for control systems description and Matlab with some Toolboxes for making experiments. We prepared a top level Modelica model which can be used as a Dymola (Modelica) block in the Matlab-Simulink environment. Actually the appropriate connectors which are compatible with other Simulink blocks had to be additionally prepared. Such the top level Modelica model is shown

in Figure 5. Two inputs (cart voltage, thread voltage) and three outputs (cart position, thread length and thread angle) were prepared. Then the Simulink environment to accept Dymola block was properly configured. This block had to be compiled within Simulink before the simulation is started. The control scheme in Simulink is depicted in Figure 7.

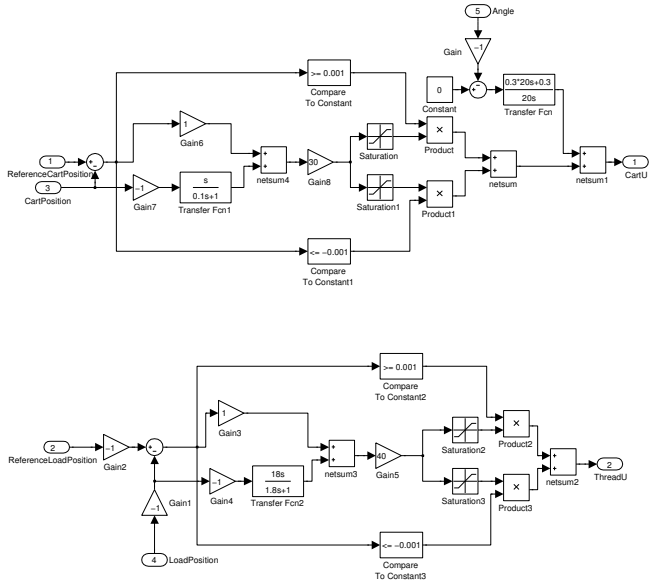

Figure 7. Control system in Simulink.

The upper part is the control scheme for cart positioning realised with PD and PI controllers

$$u_{cart}(t) = K_{Pc}\left(e_{cart}(t) + T_{Dc}\frac{de_{cart}(t)}{dt}\right) \qquad (2)$$

$$u_{angle}(t) = K_{Pa}\left(e_{angle}(t) + \frac{1}{T_{Ia}}\int e_{angle}(t)dt\right) \qquad (3)$$

where u_{cart} and u_{angle} are the appropriate control signals which influence the position of the cart in order to minimize errors between the cart reference position and actual position (e_{cart}) and thread reference and actual angle (e_{angle}). So this part of controller has 3 inputs: the reference value of the cart position (`ReferenceCartPosition`), the actual position (`CartPosition`) and the angle of the pendulum system (`Angle`). So the left part of the structure is described with Eq. 2. PD control was chosen as the process already has the integral behaviour. To realise the real behaviour the output of the PD controller was limited, so that the positive values can change at the interval $1,28V$ - $3V$ and negative values at $-1,8V$ and $-3V$. The behaviour of the model was improved by forcing the output of the controller to $0V$, when the position error was in the range of $\pm 0,001m$. At the right upper part of the scheme the PI control action (Eq.3) was superadded. Namely the influence to the cart position is the only possibility to influence the pendulum angle, which has the reference value 0°.

The lower part is the control scheme (Figure 7) handles the length of the thread. Here the PD controller was used

$$u_{thread}(t) = K_{Pth}\left(e_{thread}(t) + T_{Dth}\frac{de_{thread}(t)}{dt}\right) \qquad (4)$$

where u_{thread} is the appropriate control signal which influences the position of the length of the thread in order to minimize the error between the thread reference position and actual position (e_{thread}). Again the PD controller is used as the process itself has an integral character. The reference length of the thread is in Figure 7 signed with `ReferenceLoadPosition`), and the actual length with *CartPosition*. To match the real behaviour the output of the PD controller is limited, so that the positive values can change at the interval $3V$ - $5V$ and negative values at $-2,8V$ and $-5V$. The behaviour of the model was improved by forcing the output of the controller to $0V$, when the position error was in the range of $\pm 0,001m$.

The parameters of all three controllers are shown in Table 2.

Table 2. Controller parameters

K_{Pc}	T_{Dc}	K_{Pth}	T_{Dth}	K_{Pa}	T_{Ia}
30	1	40	18	0,3	20

Figure 8 depicts the overall Simulink scheme, where the control structure (Figure 7) was realised with a Simulink subsystem `Control system` and the physical part of the laboratory set-up with Modelica model `DymolaBlock`, which has to be compiled before simulation.

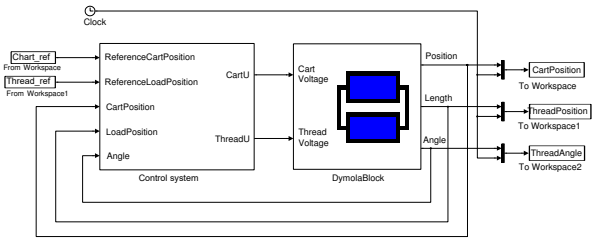

Figure 8. Overall scheme in Simulink with Modelica block.

5 Experiments

It is well known that the model validation is the most important part of each modelling procedure. It is based on comparisons between real measurements and simulation results. Many open loop and closed loop experiments were performed.

5.1 Open loop experiments

The basic validation was performed in open loop with the DC voltage inputs as shown in Figure 9. These inputs

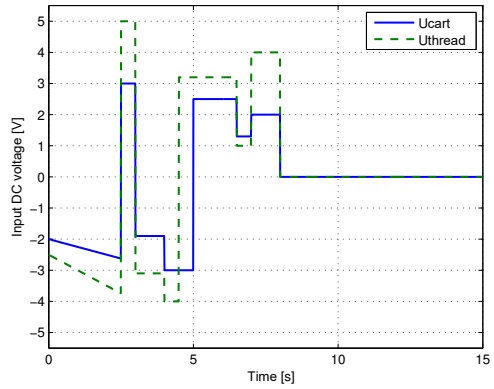

Figure 9. DC motor voltages for open loop experiments.

influenced the real loading bridge set-up and the model. According to several experiments some parameters were tuned and some procedures, which were already commented in the modelling section, were performed. The validation results presented in Figure 10 show that the model behaviour is satisfactory. It appropriately describes the movements of the cart and not so good the lowering and lifting of the weight. Especially it is difficult to tune the angle of the pendulum, where the basic frequency is properly modeled, however real measurements contain also higher frequency components.

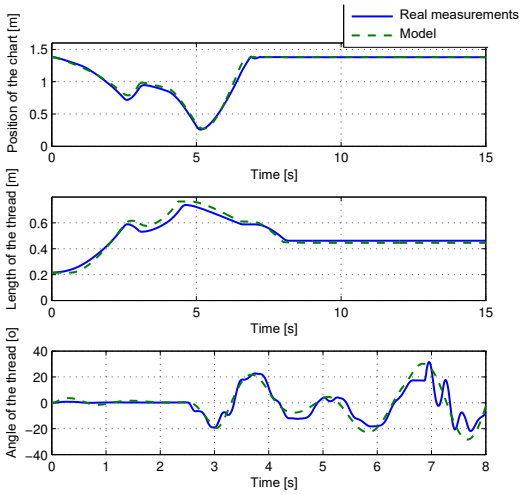

Figure 10. Open loop experiment: comparison of model outputs and corresponding real measurements.

5.2 Closed loop experiments

Figure 11 shows the position of the cart, the length of the thread and the angle of the thread in a closed loop experiment. The same reference signals were applied to the real set-up and the model and the control systems were of course identical. We can notice that the behaviour of the model is reasonable also in the closed loop. The largest deviations are again at the pendulum part, especially with its oscillations.

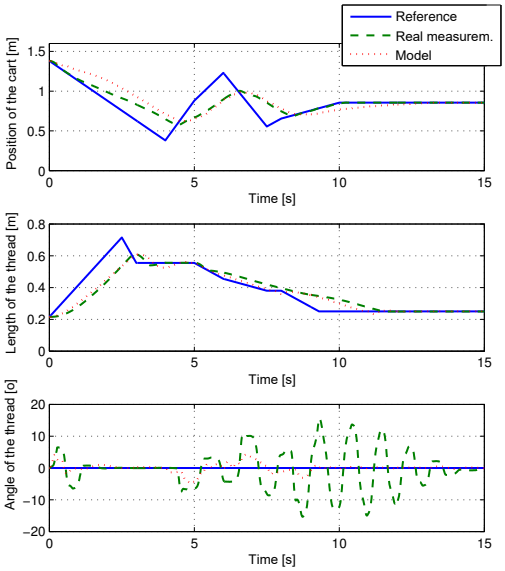

Figure 11. Closed loop experiment: reference signals, model outputs, real measurements.

As the basic goal of the control of the crane is that the load tracks the appropriate trajectory, we also present it in *xy* plane, where *x* and *y* are coordinates of the load. Figure 12 shows the reference trajectory, the trajectory of real measurements and the simulation trajectory.

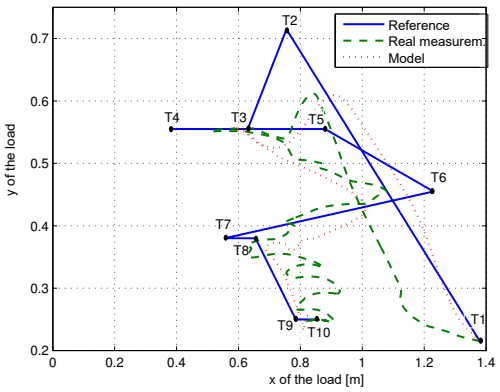

Figure 12. Presentation of trajectories in *xy* plane.

The reference trajectory is defined with points *T1* to *T10*. It has to be mentioned that controller parameters were not strictly optimised as the emphasise was given to the modelling part.

6 Conclusions

The laboratory set-up Loading bridge AMIRA 600 is a very efficient equipment for control education enabling interesting modelling and control courses. The models were previously mostly developed in Matlab-Simulink environment where all balance equations have to be precisely specified. In this investigation we tried to develop an us-

able model without equations, with OO approach using Modelica language and Standard Modelica libraries with mechanical components in Dymola environment. However some difficulties in modelling were later solved with causal approach, which was in Modelica implemented with the Block library showing that the combination of different approaches often gives better solutions. Anyway, such model much better preserves the physical modelling structure, what is important in the education but also in better understanding when model users are not modelling experts.

The experiments results confirm that the model reasonably describes the real system. The worst part of the model is the presentation of the pendulum angle where some higher frequencies also appear in real experiments. Additional efforts will be also devoted to the control system (optimisation of parameters, new control strategies, etc.).

Acknowledgements

The work described in this paper was conducted within InMotion project, co-funded by the Erasmus+ Programme of the European Union, No. 573751-EPP-1-2016-1-DE-EPPKA2-CBHE-JP.

References

Amira. *Documentation Amira PS600 V2.0; Laboratory Experiment Loading Bridge*. Amira GmbH, 2001.

Dymola. *Dymola, Dynamic Modeling Laboratory, User Manual, Volume 1*. Dassault Systèmes, Ideon Science Park, Lund, Sweden, 2015.

P. Fritzson. *Principles of Object Oriented Modeling and Simulation with Modelica 2.1*. IEEE Press, John Wiley & Sons, Inc., USA, 2004.

M. Hančič, G. Karer, and I. Škrjanc. Modeling, simulation and control of a loading bridge. *SNE Simulation Notes Europe*, 25(3–4):165–170, 2015.

Modelica Association. *Modelica Specification, version 3.2*, 2010. `http://www.modelica.org/documents/ModelicaSpec32.pdf`, (accessed May 5, 2016).

C. B. Moller. *Numerical computing with Matlab*. SIAM, Phiadelphia, USA, 2004.

OpenModelica. Open Source Modelica Consortium, 2012. `http://www.openmodelica.org`, (accessed May 5, 2016).

Simulink. *Simulink, Dynamic System Simulation Software, Users manual, R2014a*. Natick, MA, USA, 2014.

A. Sodja. *Object-oriented modelling and simulation analysis of the automatically translated models*. PhD thesis, Faculty of Electrical Engineering, University of Ljubljana, 2012. `http://msc.fe.uni-lj.si/Download//Zupancic/Dissertation_Anton_Sodja.zip`, (accessed May 5, 2016).

B Zupančič and A. Sodja. Computer-aided physical multidomain modelling: some experiences from education and industrial applications. *Simulation modelling practice and theory*, 33(6):45–67, 2013.