A Modelica Approach to ARGESIM Benchmark ‘Crane and Complex Embedded Control’ (C13R) using the Simulator Dymola

Alexander Schiftner, Vienna University of Technology; *aschift@osiris.tuwien.ac.at*

Simulator: *Modelica* is a new modern standard for defining dynamic models in an a-causal manner. *Dymola* is an simulator being able to process Modelica model descriptions. Modelica’s textual frame offers constructs for defining models in an object-oriented language, whereby implicit structures may be used. Basic models for typical dynamic behaviour may be embedded into graphical blocks and compiled as Modelica Library (some freely available; WWW.MODELICA.ORG). The simulator *Dymola* comes with comprehensive Modelica application libraries and offers advanced features for time domain analysis (index reduction methods, solvers for DAEs, sophisticated state event handling, etc.). At experiment level, Dymola comes with limited features - here a MATLAB interface may be used.

Model. The nonlinear crane was modelled using the *Multi-Body Library* of Modelica 2.2. Only graphical blocks representing mechanical devices have to be connected following the physical relations (Figure 1): a prismatic joint, in parallel with a damping element are coupled with the mass element Car - defining the car movement; the car mass is coupled via a swiwel joint and a bar element with the mass Load - defining the swinging load. Dymola translates this graphical model by symbolic transformations into an implicit model (DAE model). The resulting nonlinear equations are mathematically equivalent to the given nonlinear implicit model (different states).

It is not necessary to transform this implicit model to an explicit one, or to add terms for braking the implicit equations, because Dymola does time domain analysis with DAE solvers.

The linear crane model and the motor model were implemented using the *Modelica Block Library* (transfer functions and blocks like in SIMULINK for causal modelling; Figure 2). All digital controller actions and sensor actions were modelled textually and embedded into an Controller and Diagnosis block (Figure 3), to be linked with inputs, with motor model and with nonlinear crane model (Figure 4).

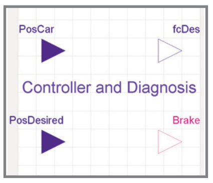

Fig. 3: Selfdefined block, embedded textual control

Modelica’s function `sample` and the `pre(.)` function (giving the previous sampled value of the argument) were used for modelling the digital controller with a fixed `sample_time` of 10 ms:

```
algorithm
  when sample(0, sample_time) then
    q := (M - d*c)*pre(q) +
        + b*pre(fcDesired) + d*pre(PosCar);
    y := h*q; u := k*PosDesired - y[1,1];
    fcDesired := max(min(u, ForceMax),
                     -ForceMax);
  end when;
```

A - Task: Nonlinear vs. Linear Model. Linear and nonlinear model were simulated independently until $t = 2.000$ s (steady state), the inputs were modelled as table functions for $f_c^{Desired}$ and f_d. Final values for x_l (Table 1) show very small differences.

Dest	x_l nonlinear	x_l linear	difference
-750	294.041	294.075	-0.034
-800	0.008	-0.005	0.013
-850	-294.112	-294.096	-0.016

Table 1: Steady state differences of nonlinear and linear calculated positions of load.

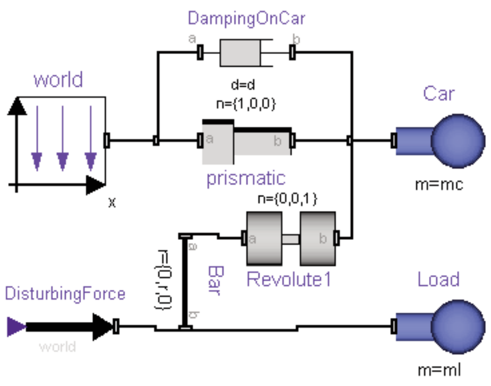

Figure 1: Nonlinear crane model implemented with Modelica MultiBody Library.

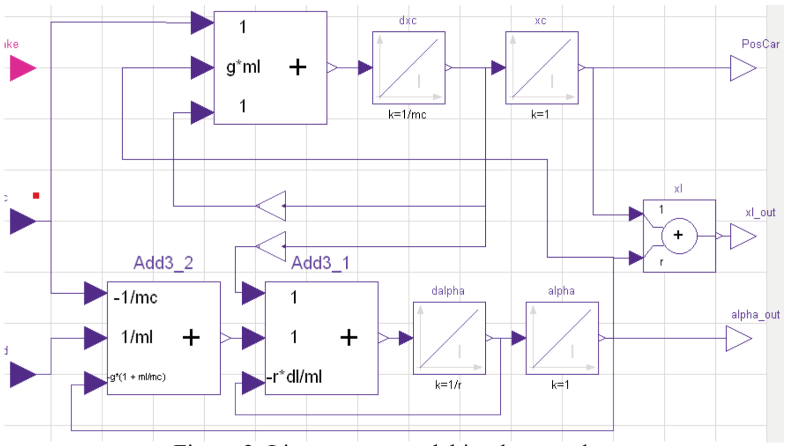

Figure 2: Linear crane model implemented with Modelica Block Library.

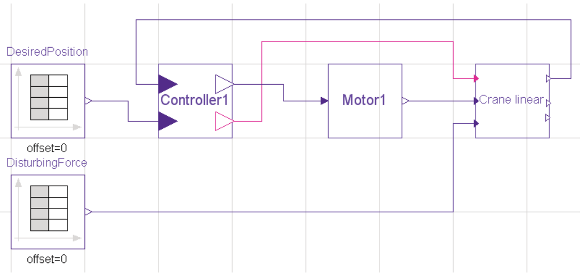

Figure 4: Inputs, controller model, motor model and nonlinear crane model connected to overall control model.

B - Task: Controlled System. Digital control is extended by brake condition, to be checked in every cycle of the digital control. A variable `ts`, representing the time since the brake condition holds, is updated accordingly and used for controlling the brake:

```
when sample(0, sample_time) then
 q := (M - d*c)*pre(q) + b*pre(fcDesired)
                    + d*pre(PosCar);
 y := h*q; u := k*PosDesired - y[1,1];
 ts := if (abs(vc) < BrakeCondition) then
          pre(ts) + sample_time else 0;
 Brake := if(ts>=3) then true else false;
 fcDesired := if Brake then 0 else
          max(min(u, ForceMax), -ForceMax);
```

Brake action is implemented by defining a conditional friction coefficient in the nonlinear crane model:

```
dc_var = if Brake then dc_Brake else dc;
```

Change of set point `PosDesired` and 1s - disturbances `fd` were modelled by table functions (Figure 4). Simulation results for the given scenario in Figure 5 show:

i) reliable control action (x_c and x_l close together),
ii) fast deflection of angle α at set point changes,
iii) brake action at $t = 35$ s caused by control treshold, followed by immediate break release caused by new setpoint at $t = 36$ s, and
v) significant deflection of angle α caused by disturbance on load at $t = 42$ s, and as consequence a bigger deviation of x_c and x_l ,
vi) but return to steady state (reached at t = 50 s).

C - Task: Controlled System with Diagnosis. A separate `when`-clause in the digital controller handles the emergency stop:

```
algorithm
 when ((PosCar>PosCarMax) or (PosCar<PosCarMin))
      then EmergencyStop := true; end when;
 when sample(0, sample_time) then .......
  ts := .......;
  Brake := if (EmergencyStop or (ts >= 3))
          then true else false;
  fcDesired := .....;
 end when;
```

Results for the diagnosis scenario in Figure 6 show:

i) - v) until $t = 46$ s same behaviour as in Task B,
vi) second disturbance at $t = 46$ s in opposite direction causes fast increase of angle α, followed by dropout of x_l and x_c, and consequently
vii) emergency stop as x_c reaches `PosCarMax` - the break fixes the car, but angle α and x_l oscillate

Resume: This Modelica / Dymola solutions makes use of different modelling paradigms. The linear dynamics are modelled by Modelica's causal Simulink-like library, nonlinear dynamics by Modelicas a-causal mechanical library. Digital control and sensor actions are implemented textually into Dymola blocks using dicrete features, and graphically composed to digital control. The nonlinear implicit equations are solved by Dymola's implicit solver, and the complex scenarios are modelled by input table functions, simulating different set points and disturbances over time.

Corresponding Author: Alexander Schiftner,
Vienna University of Technology / ARGESIM,
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
aschift@osiris.tuwien.ac.at

Received: November 28, 2006
Revised: January 20, 2007
Accepted: February 25, 2007

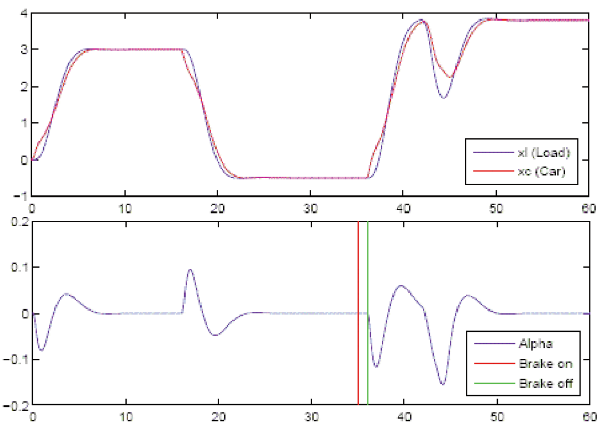

Figure 5: Time domain results for controlled model, with brake action.

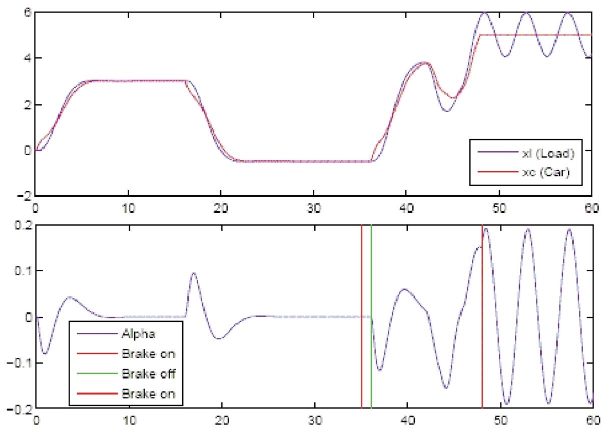

Figure 6: Results for diagnosis experiment (the second brake-on event is due to an emergency stop).