Comparison 11 – ACSL

Full Hybrid Model Approach - Environment Level

ACSL is a widely used, compiler-based simulation language for continuous models with textual and graphical model description. It provides explicit and implicit integration algorithms and (beside others) event handling features. ACSL Math is a convenient experimentation environment for ACSL with numerous analysis and graphical tools. It is based on MATLAB-like syntax. Switching between different simulation models is an easy task with the ACSL Math `load` command. This feature can be used to implement a "Full Hybrid Model Approach" by loading and unloading models with different numbers of states depending on actual conditions during a simulation run.

Model description (Task a): The mathematical model for the three-axis SCARA robot has three degrees of freedom (DOF). Consequently, there is one 3-DOF-model (full model), three 2-DOF- and three 1-DOF-models, i.e. altogether 7 different models.

The following abbreviated `DERIVATIVE` Sections show the essentials of two different reduced model descriptions. Example 1 is an implicit 2-DOF-model for a motion of the two revolute joints. Example 2 is an explicit model for the vertical DOF `q3` of the robot hand.

```
DERIVATIVE ! Begin Model scara_110_imp
ma11 = th1+2*th2*c2+th3; ma12 = ...
b1 = t1+th2*(2*dq1*dq2+dq2**2)*s, b2 = ...
residdq1 = ma11*ddq1 + ma12*ddq2 -b1
residdq2 = ma21*ddq1 + ma22*ddq2 -b2
dq1, ddq1 = IMPLC(residdq1, dq10)
dq2, ddq2 = IMPLC(residdq2, dq20)
q1 = INTEG( dq1, q10); q2 = ...
END ! Derivative Model scara_110_imp
DERIVATIVE ! Begin Model scara_001_exp
b3 = t3-m3L*g
ddq3 = b3 / ma33
dq3 = INTEG(ddq3, dq30)
q3 = INTEG( dq3, q30)
END ! Derivative Model scara_001_exp
```

Since ACSL allows implicit and explicit model descriptions the full model and the reduced models can be programmed both ways and can be used even alternately when switching models.

Point to Point Control (Task b): Servo motors and controllers can be easily implemented by standard modelling features of ACSL. To take advantage of the hybrid model approach, the position errors have to be monitored and evaluated by the ACSL `TERMT` command. After reaching the target position for a particular joint, the simulation is halted and control is returned to ACSL Math. Depending on which axis caused the termination the matching reduced model is loaded and initialized. Simulation then continues for the drives that are still in motion. Figure 1 shows the time history for the joint angles, composed from results of three subsequently executed models.

Computation time is reduced to approximately 45% by this approach since the reduced models are executed faster due to the smaller number of DOFs.

```
load @file=scara_111_exp @format=model
 !!prepare t,q1,q2,q3
 start
.......
if (halt == 100 & explicit)
  load @file=scara_011_exp @format=model
elseif (halt == 010 & explicit)
  load @file=scara_011_exp @format=model
 .......
end
```

Model Description	Implicit		Explicit
Integr. Algorithm (Stepsize 0.005 s)	RK-4 `IALG=5`	DASSL `IALG=10`	RK-4 `IALG=5`
Rel. CPU-time Full Model	1.0	0.86	0.12
Rel. CPU-time Subset Models	0.45	0.39	0.05

Computation times on a HP715/100, ACSL Vers. 11

Obstacle avoidance (Task c): To detect a state event the SCHEDULE operator is used in ACSL which starts an iterative state event locating routine and finally executes a DISCRETE Section. For collision avoidance a generic SCHEDULE command is used in the ACSL model description and the actual state event to be checked for is selected via the index variable `ichk`.

```
chkvar(1)=d-dcr; chkvar(2)=h
SCHEDULE event .XZ. chkvar(ichk)
```

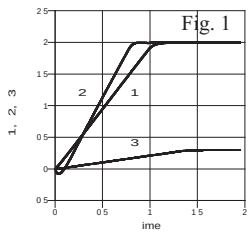
If the value of the checked variable `chkvar(ichk)` crosses zero the DISCRETE Section `event` is triggered and the simulation run is terminated:

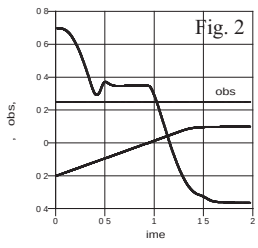
```
DISCRETE event ; TERM(.true.) ; END
```

On return from a simulation run an ACSL Math script has to check the termination condition and to start suitable script files, either to load another (reduced) model or to take precautions for obstacle avoidance.

```
while (time < tend)
 !! START
 collect_data  % script to save prepared data
 if (halt  > 0)
  select_model % script to load new model
 elseif (h  < 0)
  handle_event % script to avoid obstacle
 end
end
```

Figure 2 was plotted using ACSL Math. It shows that the x-position of the tool tip does not cross the obstacle border line until the tool tip height has reached a positive height above the obstacle.





Horst Ecker, Institute for Machine Dynamics and Measurements, TU Vienna, Wiedner Hauptstraße 8-10, A-1040 Vienna/Austria, email: horst.ecker@tuwien.ac.at