Ecole Centrale de Nantes
EMARO-ARIA M1

LAB 4 : Co-Simulation Control of Robot Arm Dynamics
in ADAMS and MATLAB

Developed by MSC Company, ADAMS software provides powerful model and simulation environment and has strong analysis function about kinematics and dynamics, which is popular in mechanical project fields in the world. While, MA TLAB becomes indispensable utility software during scientific research owing to its powerful computed function, visualization of programming and calculating results and high efficiency of programming. For control system design, control toolbox in ADAMS is incomplete; however, SIMULINK section of MATLAB makes up this drawback. If combine two software together, in other words, combine mechanical system simulation with design of control system together, advantages of each software are utilized and electromechanical co-analysis is done. By that way, we can build complex control scheme, in the meantime observing motor process of target object, which is beneficial to research of complex electromechanical system.

Co-simulated method takes advantages of two software, with enhancing dynamic performance of robot arm, improving efficiency, reducing the cost and saving time. For the complex control system, it is a good solution selected.

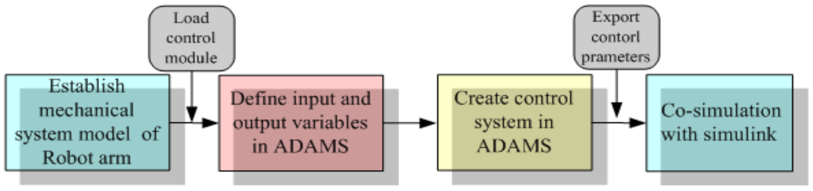

Load control module
Export contorl prameters
Establish mechanical system model of Robot arm
Define input and output variables in ADAMS
Create control system in ADAMS
Co-simulation with simulink

Figure 1: Co-simulation flow chart of ADAMS and MATLAB.

1 Objective

The main objective of the present lab is to study the bahaviour of a Kuka LWR Robot model in the Matlab environment by applied some classical linear and non-linear robotic control (PID, Computed Torque) in the joint space and/or the task space and then to compare it with results obtained with Adams.

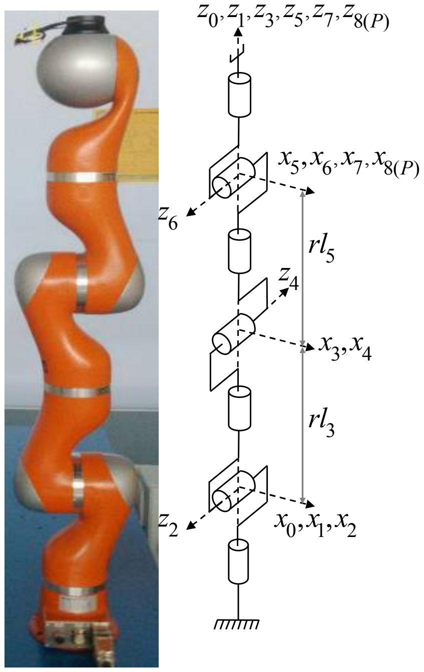
The Adams model used for this lab is named "`KukaSolids1.bin`".

2 Establishment of Kuka LWR robot dynamics model

The **IDM** (**I**nverse **D**ynamic **M**odel) of a robot calculates the torques τ as a function the motor positions ($\mathbf{q}$), velocities ($\dot{\mathbf{q}}$) and accelerations ($\ddot{\mathbf{q}}$). It can be obtained from the Newton-Euler or the Lagrangian equations [1].

2.1 Description of the robot and its kinematics

The Kuka LWR[1] (see Fig. 2) robot has a serial structure with $n = 7$ rotational joints. Each motor has encoder which measures the motor position.



$z_0, z_1, z_3, z_5, z_7, z_{8(P)}$
$x_5, x_6, x_7, x_{8(P)}$
z_6
rl_5
z_4
x_3, x_4
rl_3
z_2
x_0, x_1, x_2

Figure 2: Link frames of the Kuka LWR.

The kinematics of serial robots is defined using the Modified Denavit and Hartenberg (MDH) notation [1].

In this notation, the link j fixed frame is defined such that: the z_j axis is taken along joint j axis; the x_j axis is along the common normal between z_j and z_{j+1}; α_j and d_j parameterize the angle and distance between z_{j-1} and z_j along x_{j-1}, respectively; θ_j and r_j parameterize the angle and distance between x_{j-1} and x_j along z_j , respectively; $a(j)$ denotes the link antecedent to link j.

[1]LightWeight kinematically Redundant robot.

j	σ_j	α_j	d_j	θ_j	r_j
1	0	0	0	θ_1	0
2	0	$\pi/2$	0	θ_2	0
3	0	$-\pi/2$	0	θ_3	$r_{l_3} = 0.4$
4	0	$-\pi/2$	0	θ_4	0
5	0	$\pi/2$	0	θ_5	$r_{l_5} = 0.39$
6	0	$\pi/2$	0	θ_6	0
7	0	$-\pi/2$	0	θ_7	0

Table 1: MDH Parameters of Kuka LWR robot.

2.2 Theoretical background

The complete **IDM** of the robot is given by the following relation with the consideration of motor torques and torque sensors measurements:

$$\begin{aligned}
\tau_{idm_m} &= \mathbf{diag}(\ddot{\mathbf{q}})\mathrm{I}_{am} + \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\dot{\mathbf{q}}, \mathbf{q}) + \tau_{fm} + \tau_{fl} && (1)\\
\tau_{idm_l} &= \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{H}(\dot{\mathbf{q}}, \mathbf{q}) + \tau_{fl} && (2)\\
\tau_{idm_m} - \tau_{idm_l} &= \mathbf{diag}(\ddot{\mathbf{q}})\mathrm{I}_{am} + \tau_{fm} && (3)
\end{aligned}$$

with:

$$\begin{aligned}
\tau_{f_m} &= \mathbf{diag}(\dot{\mathbf{q}})F_{vm} + \mathbf{diag}(\mathrm{sign}(\dot{\mathbf{q}}))F_{cm} + \mathit{offm} && (4)\\
\tau_{f_l} &= \mathbf{diag}(\dot{\mathbf{q}})F_{vl} + \mathbf{diag}(\mathrm{sign}(\dot{\mathbf{q}}))F_{cl} + \mathit{offl} && (5)
\end{aligned}$$

where:

- n is the number of moving links.
- $\mathbf{M}(\mathbf{q})$ is is the $(n \times n)$ robot inertia matrix;
- $\mathbf{H}(\dot{\mathbf{q}}, \mathbf{q})$ is the $(n \times n)$ vector of Coriolis, centrifugal, gravitational and friction forces/torques;
- I_{am} is the $(n \times 1)$ vector of total inertia moments for rotors and gears;
- F_{vm} and F_{cm} are the $(n \times 1)$ vector of viscous and Coulomb friction parameters of motor side;
- F_{vl} and F_{cl} are the $(n \times 1)$ vector of viscous and Coulomb friction parameters of link side;
- *offm* is the $(n \times 1)$ vector of motor current amplifier offset parameters;
- *offl* is the $(n \times 1)$ vector of torque sensor offset parameters;

The choice of the modified Denavit and Hartenberg frames attached to each link allows a dynamic model that is linear in relation to a set of standard dynamic parameters χ_{st}:

$$\tau_{idm} = \mathrm{IDM}_{st}(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q})\chi_{st} \qquad (6)$$

Where $\text{IDM}_{st}(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q})$ is the $(n \times N_s)$ jacobian matrix of τ_{idm}, with respect to the $(N_s \times 1)$ vector χ_{st} of the standard parameters. χ_{st_j} is composed of the following standard dynamic parameters of axis j:

$$\begin{aligned}
\chi^{ms}_{st_j} &= \left[\begin{array}{cccc} I_{a_j} & F_{vm_j} & F_{cm_j} & \mathit{offm}_j \end{array} \right] \\
\chi^{ls}_{st_j} &= \left[\begin{array}{ccccccccccccc} XX_j & XY_j & XZ_j & YY_j & YZ_j & ZZ_j & MX_j & MY_j & MZ_j & M_j & F_{vl_j} & F_{cl_j} & \mathit{offl}_j \end{array} \right]
\end{aligned}$$

with

$$\chi_{st_j} = \left[\begin{array}{cc} \chi^{ms}_{st_j} & \chi^{ls}_{st_j} \end{array} \right] \tag{7}$$

where:

- I_{am_j} is a total inertia moment for rotor and gears of actuator of link j (to simplify: it is named drive inertia moment);
- F_{vm_j} and F_{cm_j} are the viscous and Coulomb friction parameters of joint j (motor side);
- Offm_j is the motor current amplifier offset of joint j;
- XX_j, XY_j, XZ_j, YY_j, YZ_j and ZZ_j are the six components of the robot inertia matrix of link j;
- MX_j, MY_j and MZ_j, are the components of the first moments of link j;
- M_j is the mass of link j;
- F_{vl_j} and F_{cl_j} are the $(n \times 1)$ vector of viscous and Coulomb friction parameters of joint j (link side);
- Offl_j is the torque sensor offset of joint j;

2.3 Identification

For the **identification** of the dynamic parameters of the rigid model of the KUKA LWR robot, the Inverse Dynamic Identification Model (witch calculates the motor torques that are linear in relation to the dynamic parameters of both links and drive chains and use linear least squares technique (IDIM-LS technique)) can be used based on the measures of the motor positions and the motor currents, or the torque sensors measurements or both side data (see [2] for more details.).

2.4 Modeling validation

After identification process achieved and for the purpose to validate the modeling, we build a first simulator under Matlab/Simulink. When using this first simulator, and after downloading `lab4_Sim1.zip` from the website, `http://pedagogiev2.ec-nantes.fr`, follow the different steps:

- Go to `lab4_SIM1`;
- Open `simu_lwr_rig_essentiels.mdl` (see Fig. 3) and after analyse and explain their content (role of each box?).
- What type of data contained in the file `consignes.mat`?

- Run the program `prog_princi_simu.m` and observe the results.
- Analyse and explain these results.
- Conclude.

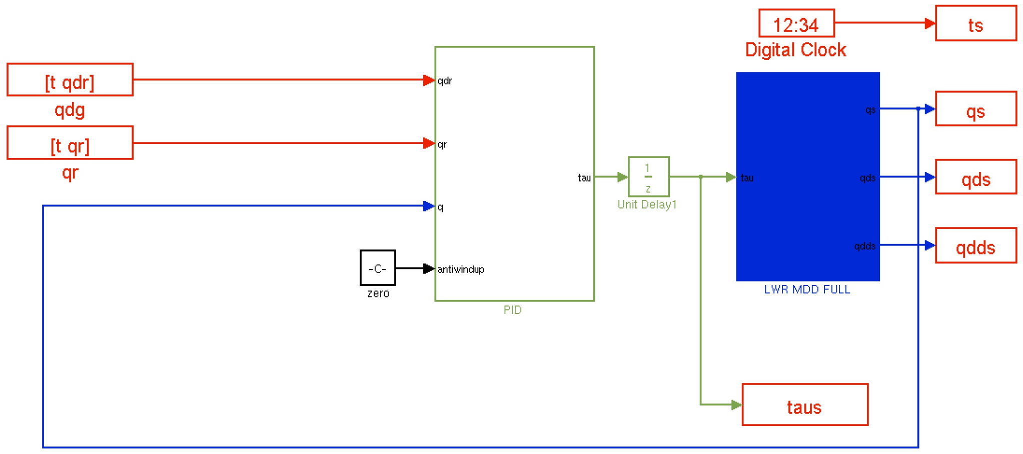


Figure 3: First simulator of the Kuka LWR robot.

3 Establishment of robot arm co-simulation model with ADAMS and Matlab/Simulink

Co-simulation with ADAMS and MA TLAB /SIMULINK means that build multi-body system in ADAMS, output parameters related to system equation and then import information from ADAMS into MATLAB/SIMULINK and set up control scheme. During calculation process, there exchanges data between virtual prototype and control program, where, ADAMS solves the mechanical system equation and MATLAB solves the control system equation. They both finish the whole control process. The flow chart of co-simulation displays in Fig 1.

3.1 Build the interface between two software

The model built in ADAMS, as a **sub-system**, need to be imported into MATLAB/SIMULINK, on which SIMULINK constructs the co-simulation system.

First, exchange data between ADAMS and MA TLAB/SIMULINK through ADAMS/CONTROL interface. Second, define 21 system variables (*Creation of the Input/Output of the plant*) which is needed in co-simulation such as:

- **input variable**: `tau1, tau2, tau3, tau4, tau5, tau6, tau7`;
- **output variable**:
  1. `q1, q2, q3, q4, q5, q6, q7` (joint positions);

2. `q1dot, q2dot, q3dot, q4dot, q5dot, q6dot, q7dot` (joint velocities).

After defining the *Input/Outputs* variables, we **export** the **Matlab model** from ADMAS/CONTROL. This will generate three files (`.m`, `.cmd` and `.adm` file) which is useful in data-exchange between ADAMS and MATLAB. In our case, these are the files : `CPU_jnt_space.m`, `CPU_jnt_space.cmd` and `CPU_jnt_space.adm`

3.1.1 Interffacing Adams with Matlab

Follow the following steps:

1. From the website, `http://pedagogiev2.ec-nantes.fr`, download the `lab4_Sim2.zip` file.
2. Run `CPU_jnt_space.m`;
3. Input command `adams_sys` into MATLAB, mechanical sub-system generates (Fig. 4 and Fig. 5).

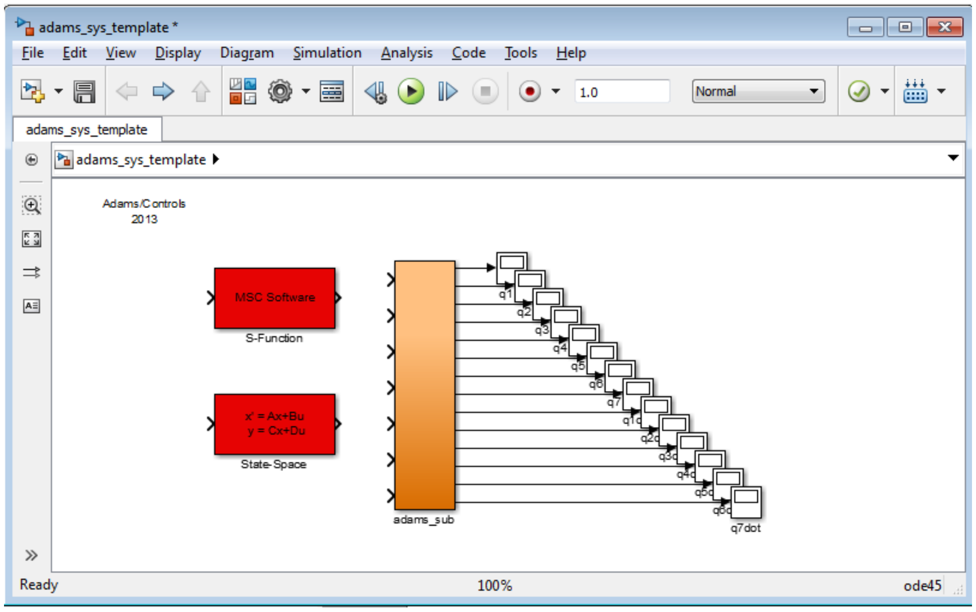


Figure 4: Adams-sys-template module.

You are now ready to design your control simulator under Matlab/Simulink and conduct the co-simulation. This is a interactive closed process, in which simulation is real-time. During the simulation, real-time interactive process is shown on the ADAMS interface.

3.2 Simulation

1. To get an idea on how to use the red block (MSC Software, Adams Plant, Fig. 6), we will start this step by a simulation example. To do this, first download the file `lwr_SV.zip`, run the simulation by executing the file `test_lwr_trq_control.m`. What type of control laws is programmed? **Observe the results and conclude**.

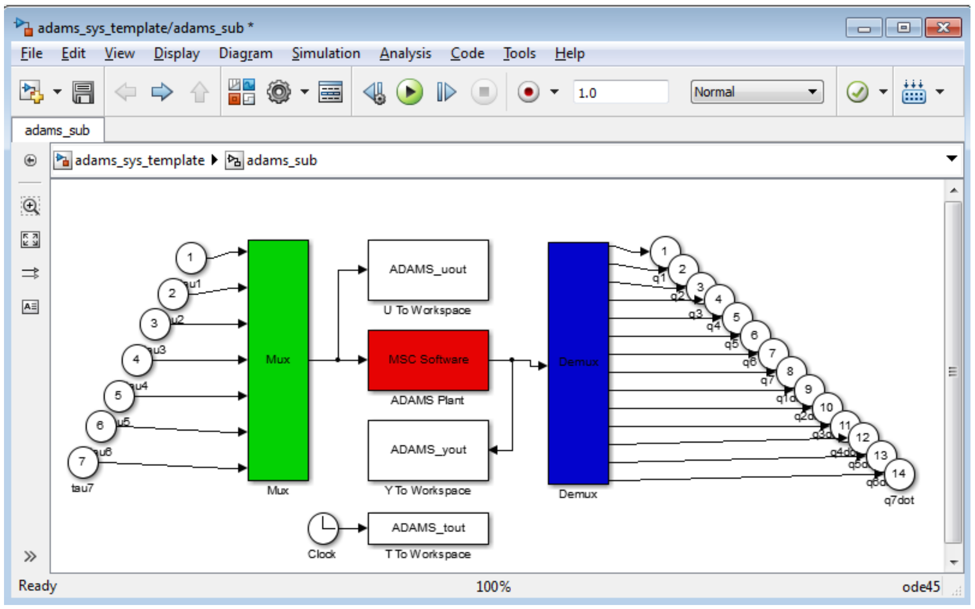


Figure 5: Adams-sub module.

2. Develop in Matlab/Simulink a simulator to control the Kuka robot (Adams model) with computed control law and in the joint space.

3. Develop in Matlab/Simulink a simulator to control the Kuka robot (Adams model) with computed control law and in the task (cartesian) space.

For the two last cases, to build your simulator, it is available to you two folder: “`Lab4Toolbox`” and “`KukaFunctions`”. The first directory contains the simulink file “`lab4toolbox.mdl`” (or `lab4toolbox.xls`) which itself contains simulink block that make calls to Matlab functions contained in “`KukaFunctions`” directory. You just need to add any files to the path.

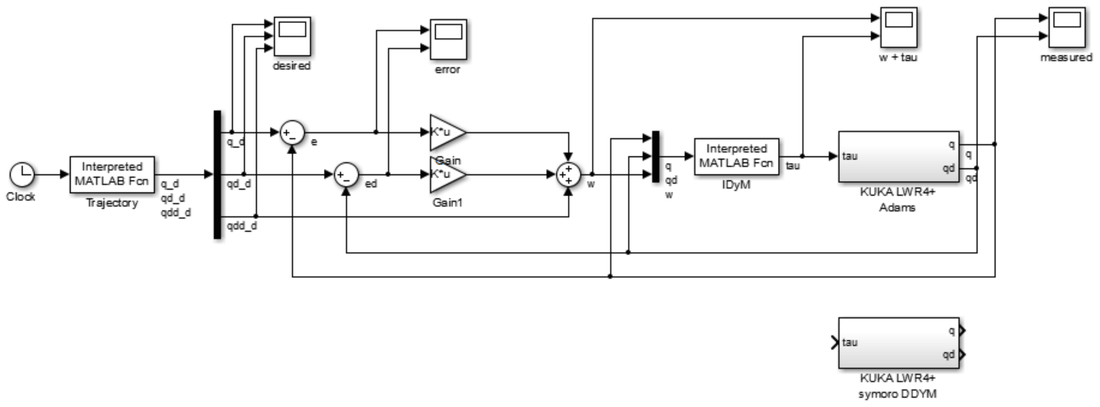


Figure 6: Second simulator of the Kuka LWR robot.

References

[1] Khalil, W., Dombre, E., 2002. *Modeling, Identification and Control of Robots*. 3rd edition. ed. Taylor and Francis Group, New York.

[2] Jubien, A., Gautier, M., Janot, A. *Dynamic identification of the Kuka LWR robot using motor torques and joint torque sensors data*. World Congress of the International Federation of Automatic Control (IFAC), Aug 2014, Cape Town, South Africa. pp.1-6.