Simulink Impedance Controller for Barrett WAM

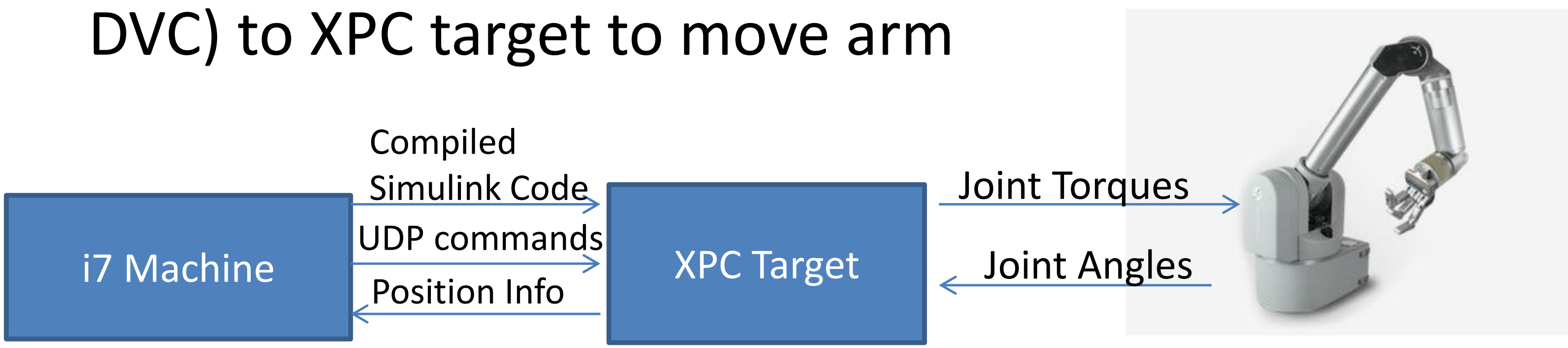
Chris Jordan

Overview

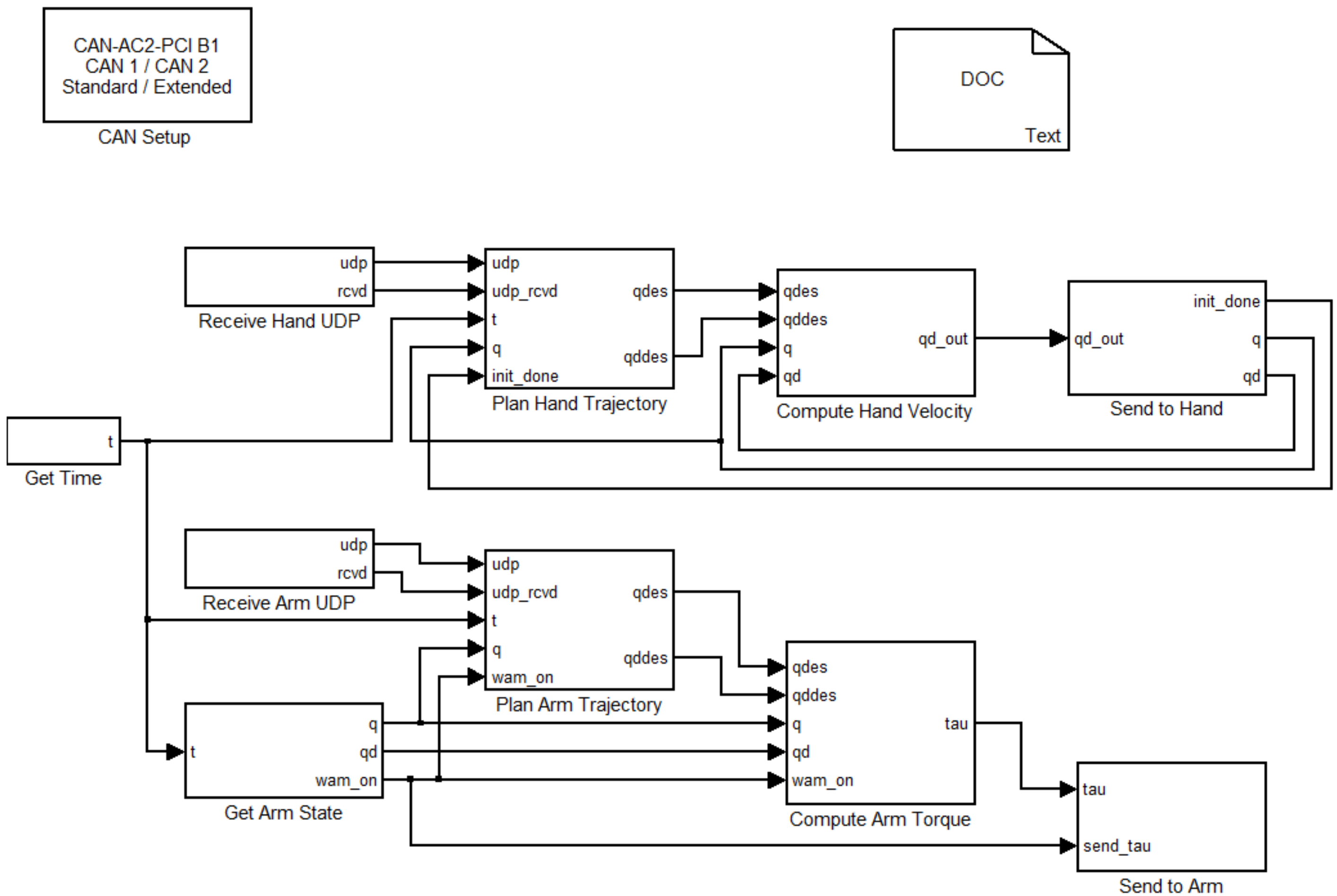
- Summary of the existing System
- Theory of Impedance Control
- Current Implementation
- Demo
- Implementation Problems

Existing System

- WAM connected to XPC Target machine using CAN communication
- Program in MATLAB Simulink (on i7 machine) and download code to XPC target to be run in real-time
- Send UDP commands from i7 machine (or potentially DVC) to XPC target to move arm

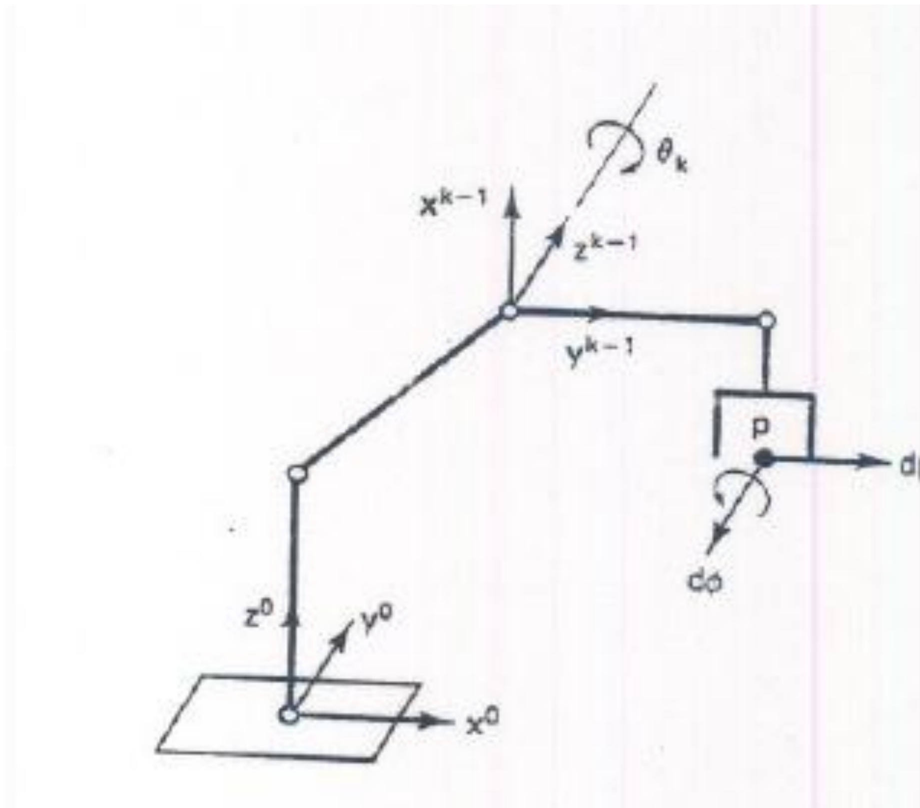


Simulink Block Diagram



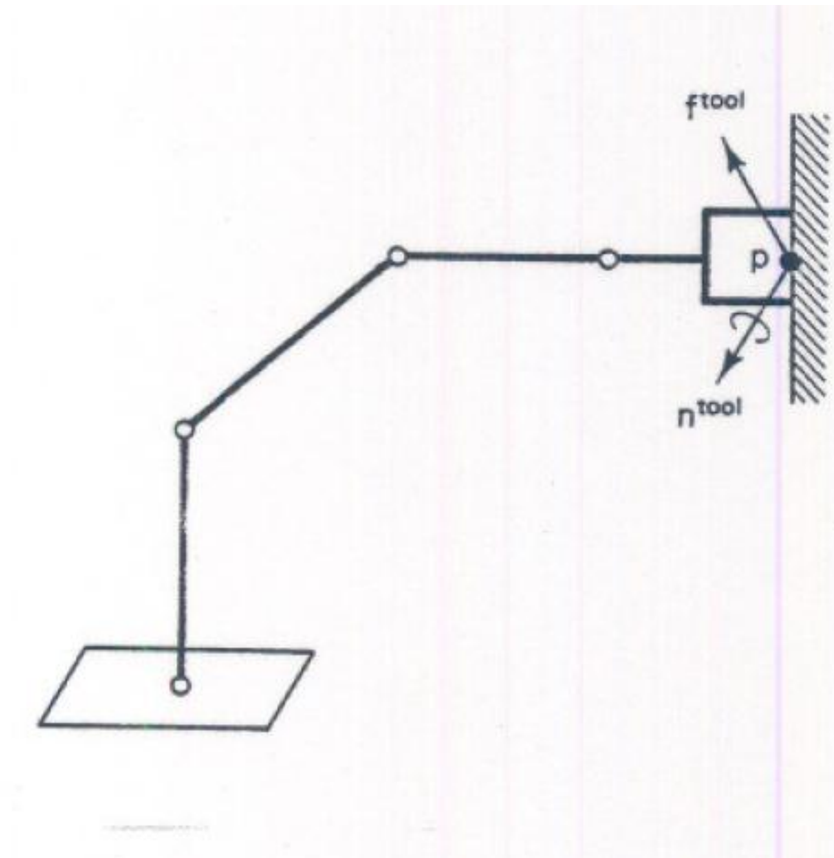
Force Controller

- Objective: Tool interacts with environment and is subject to forces/moments. Force controllers deal with how to compensate during these interactions
- Example: robotic arm writing with a pencil, or during a grasp with the object



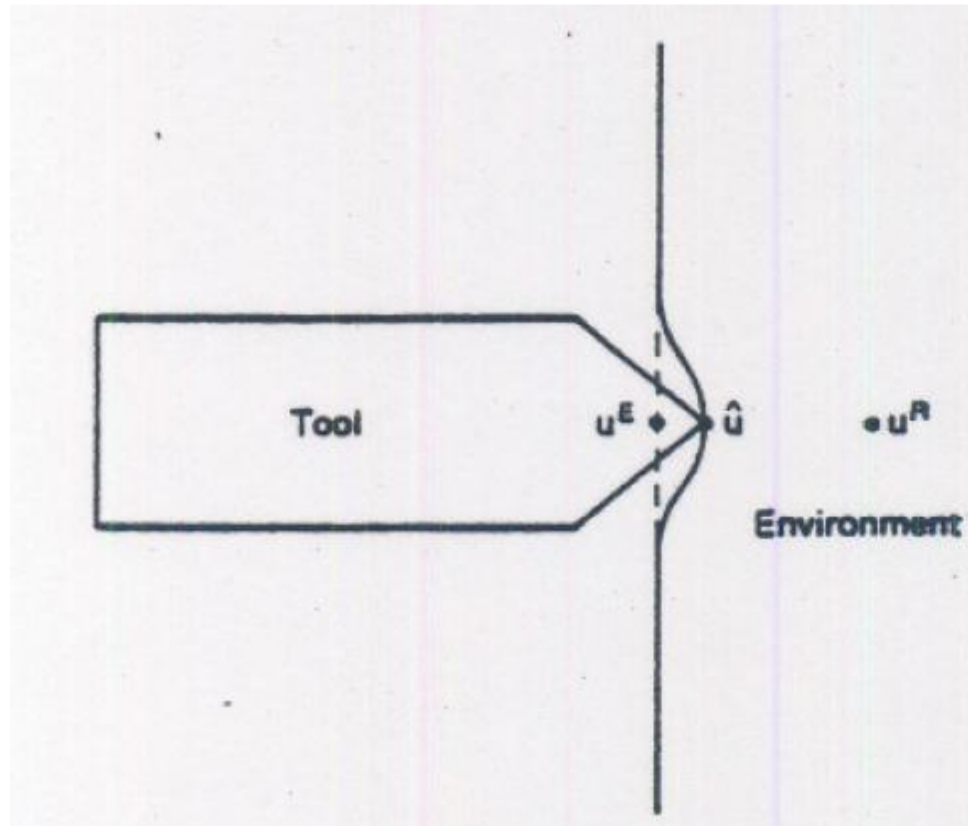
F^{tool} Matrix

- $F^{tool} = \begin{bmatrix} force_x \\ force_y \\ force_z \\ moment_x \\ moment_y \\ moment_z \end{bmatrix}$
- In base coordinates



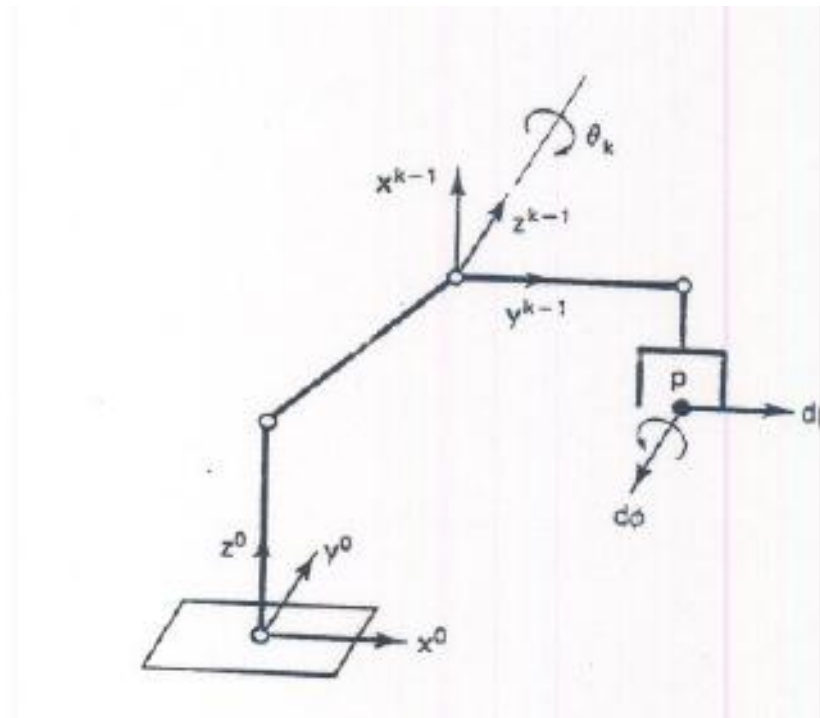
Error Signal

- $F^{tool} = H(u^e - u)$
  - Force created from environment
- $error = u^r - u$
  - =desired-actual
  - Cartesian position
  - Used by impedance controller



Derivation of Manipulator Jacobian

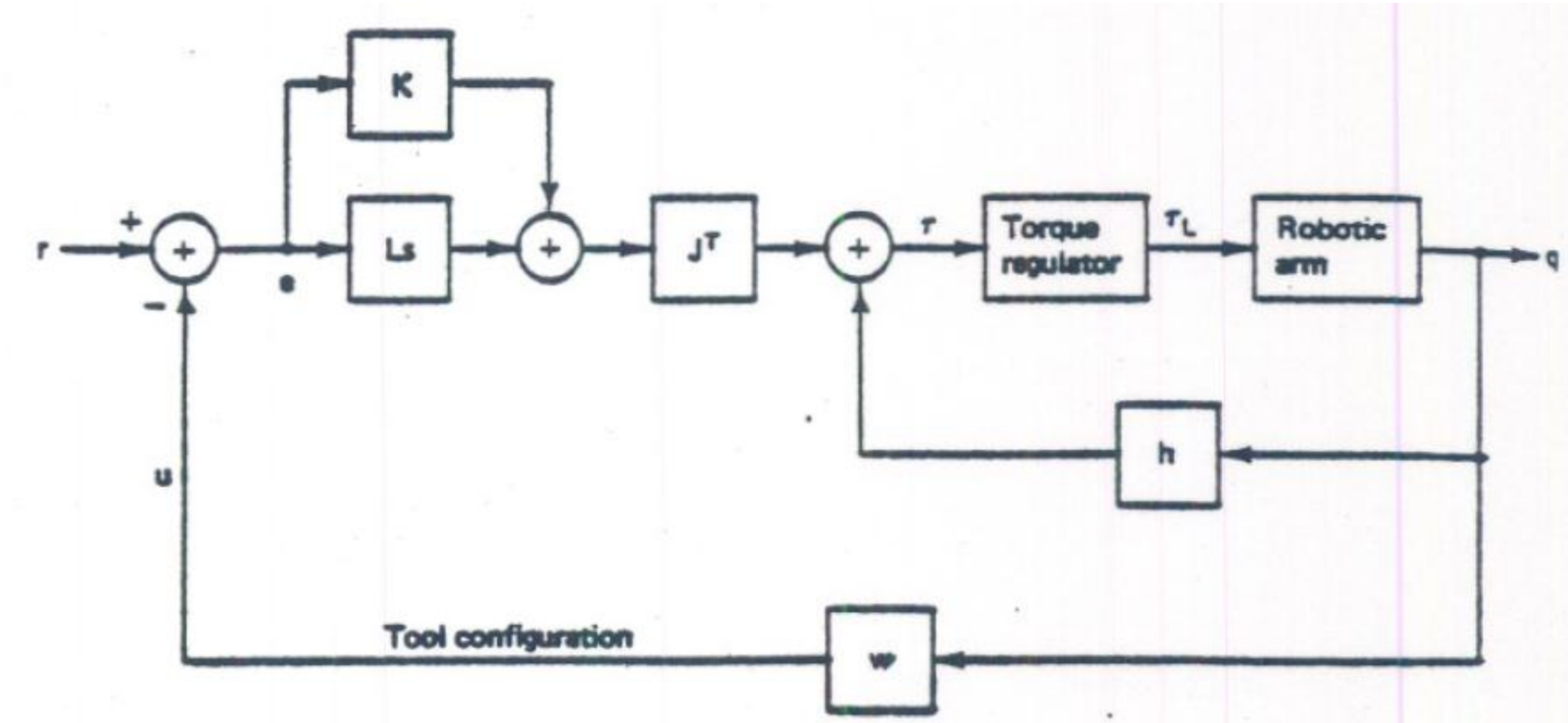
- $J(q) = \begin{bmatrix} A(q) \\ B(q) \end{bmatrix}$
  - $A(q) = \frac{dp(q)}{dq_k}$ for k=1:n
  - B(q)=third column of each rotation matrix from 0 to n-1
    - [0; 0; 0] if a prismatic joint



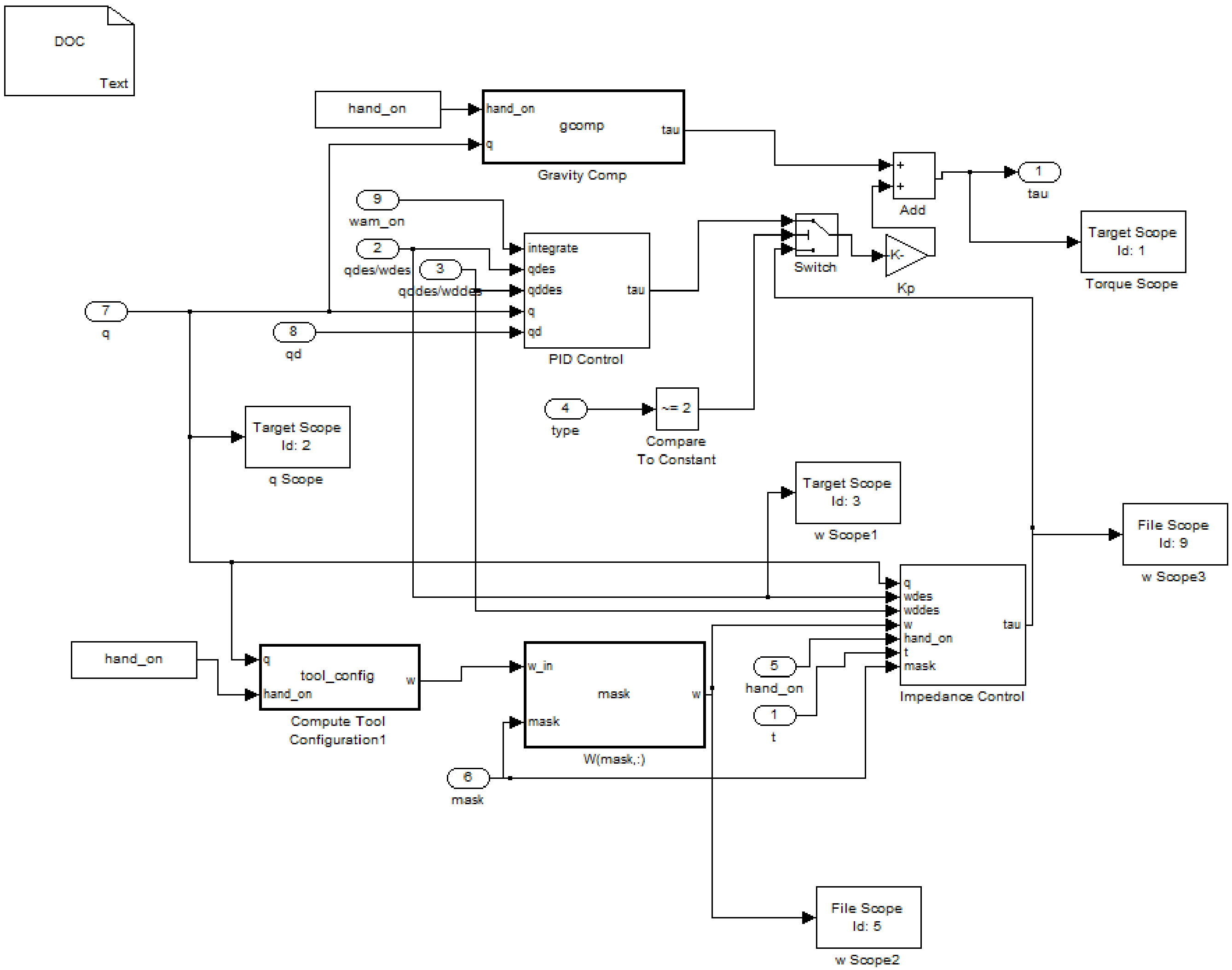
Theory of Impedance Controllers

- $du = J(q)dq$
  - Infinitesimal tool displacement= Jacobian * infinitesimal Joint displacement
- $F^{tool} = Kdu$
  - K is the spring constant (determines "Stiffness")
  - If large, the controller is similar to PD
  - Should be selected relative to the environment
- $du = e = r - u$
- $\tau = J^T(q) * F^{tool}$
  - A force at the end effector exerts these torques on each joint
- $\boldsymbol{\tau} = \boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{q}) * [\boldsymbol{Ke} + \boldsymbol{Le}'] + \boldsymbol{h}(\boldsymbol{q})$
  - Where h(q) is gravity compensation

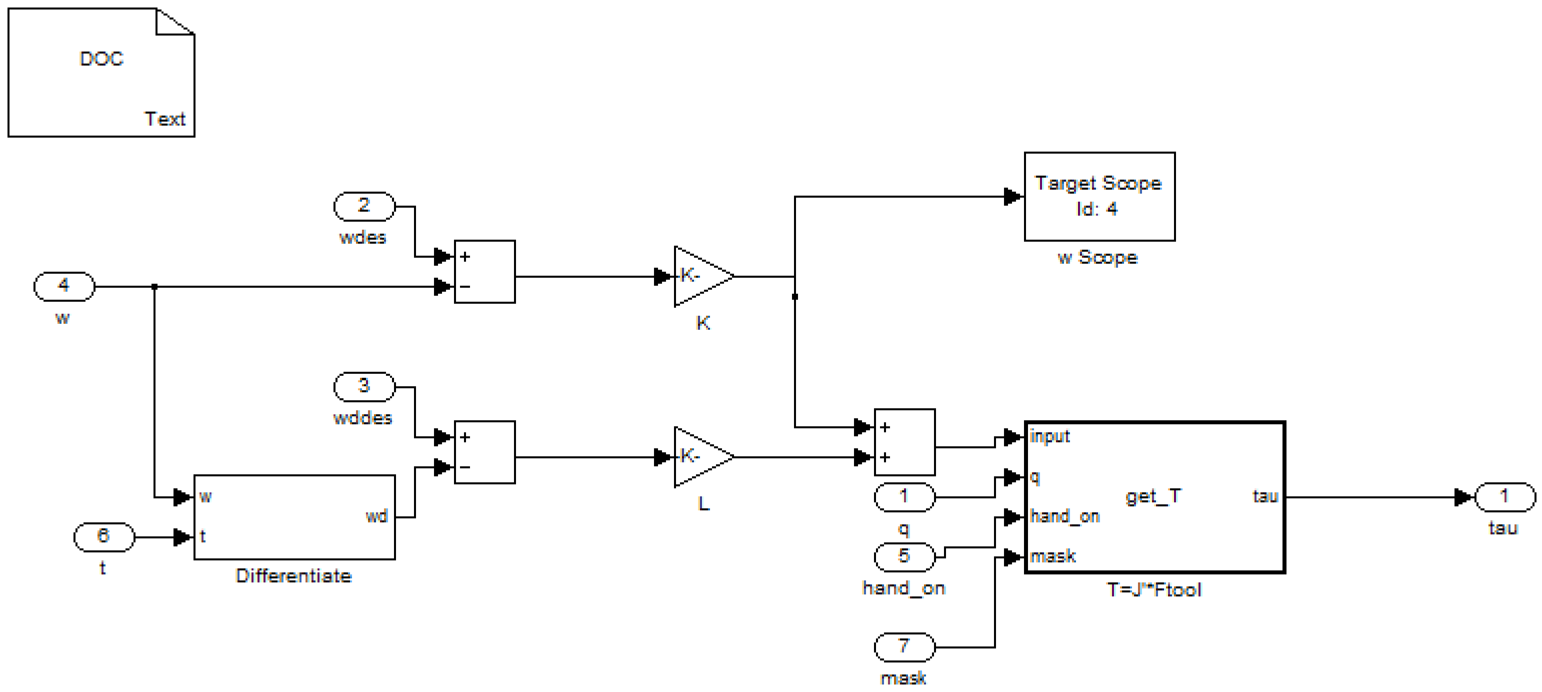
Block Diagram



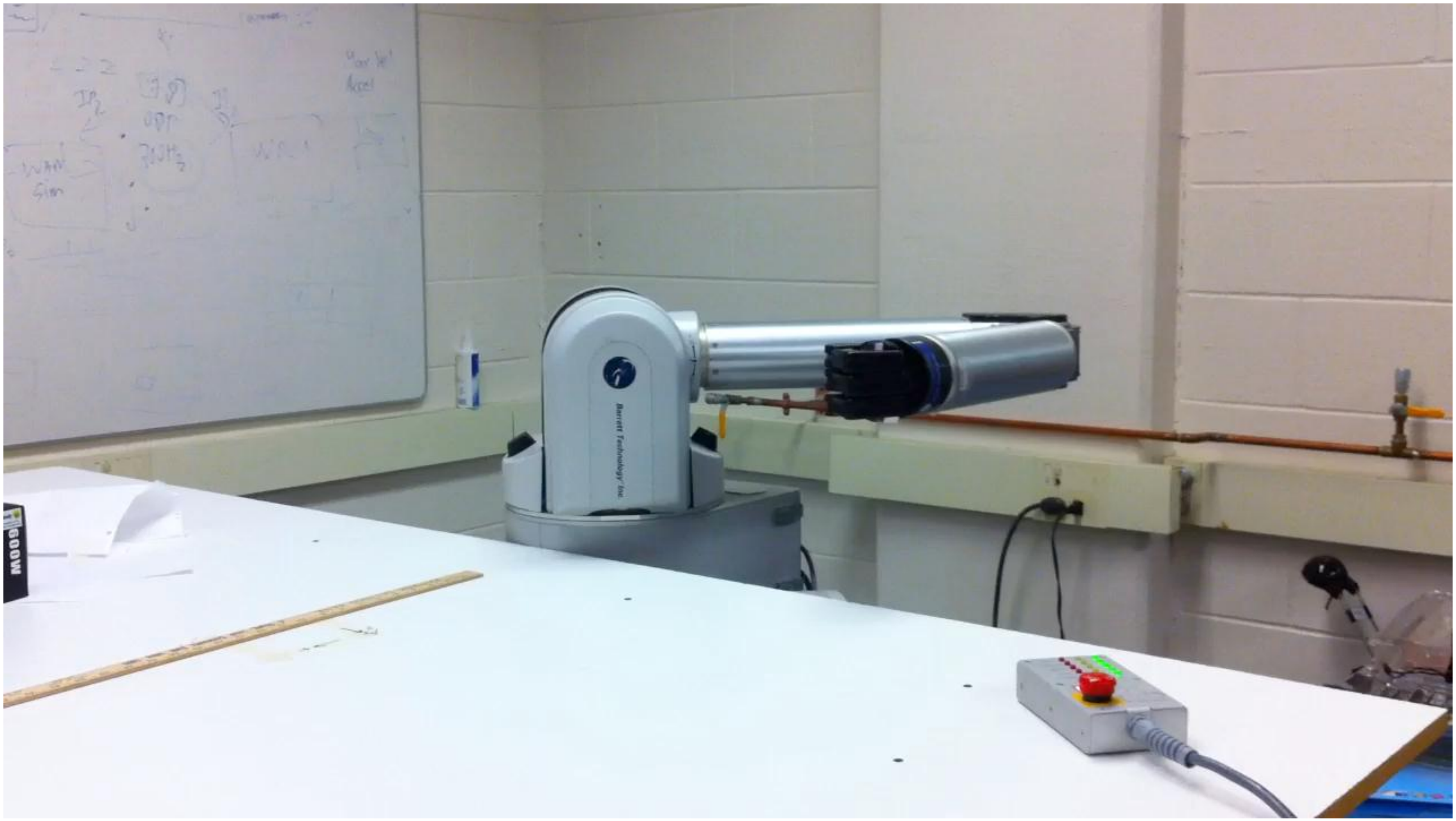
Simulink Controller



Current Implementation

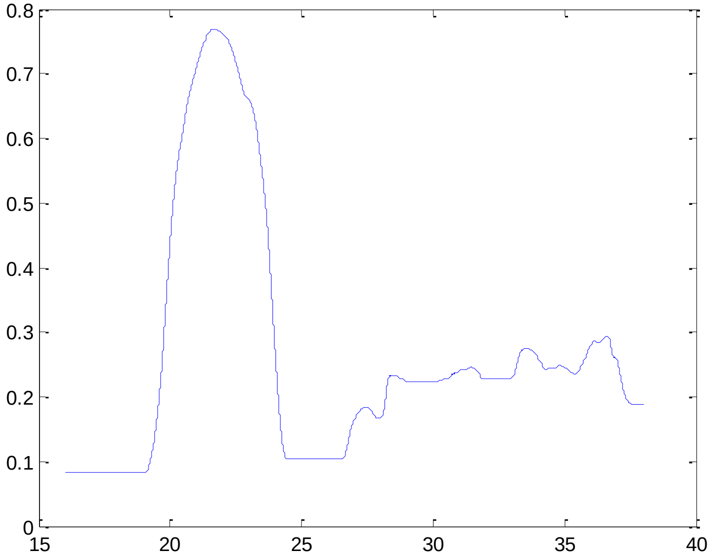


Demonstration
(stiffness gains=[50 200 200] in [x y z])

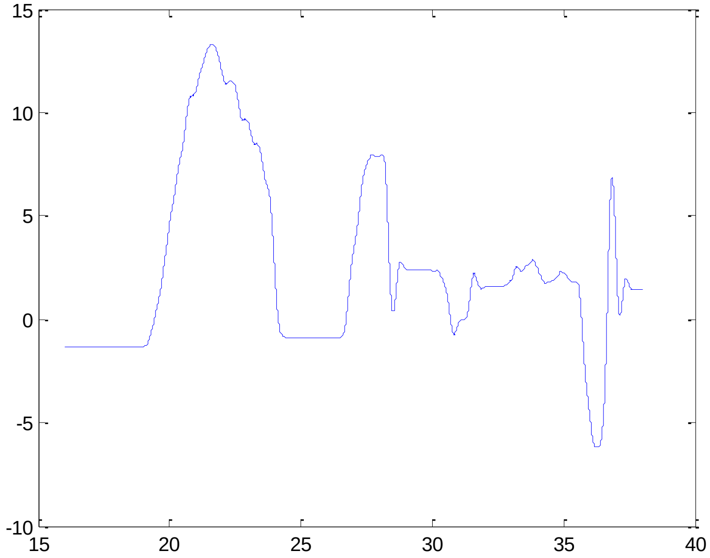


Sample Data

Position in X



Torque Joint 1



Implementation Problems

- Integration with existing system
  - Problems with the current task planner
- Orientation Support

Future Works

- Fix minor bugs
  - Integration with the task planner
  - Problem with mask implementation
- Add support for orientation
- Add ability to change stiffness and damping gains over UDP
- Debug system

Questions?