16711 Kinematics, Dynamics and Controls - HW3

Ratnesh Madaan (ratneshm), Jerry Hsiung (shsiung), Rogerio Bonatti (rbonatti)

Due: 02/20/2018

1 Main Concepts of Manipulator Kinematics

1.1 (a) Find the forward kinematics map

Let's use the product of exponentials formula to derive the kinematics map according to Figure 7

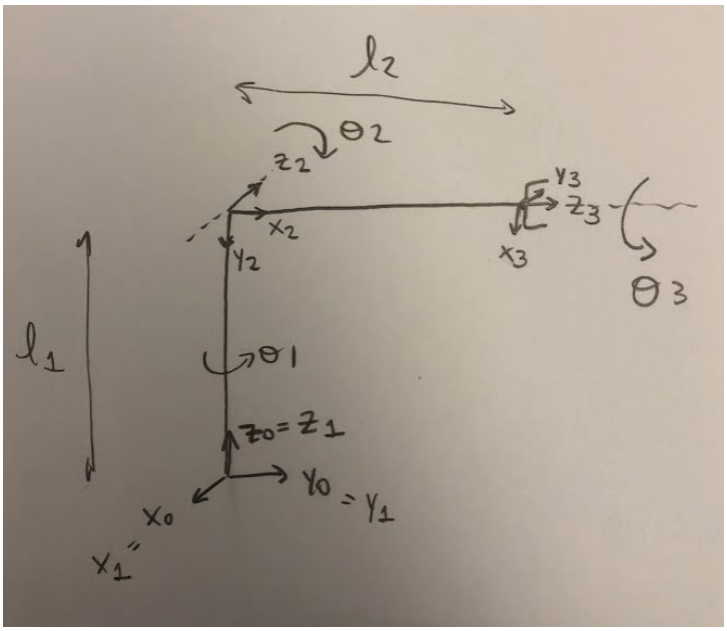

Figure 1: Frame of robot

$$g_{tool} = e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_2}\theta_2} e^{\hat{\xi_3}\theta_3} g_{st}(0)$$
$$g_{tool} = g_1(\theta_1) g_2(\theta_2) g_3(\theta_3) g_{st}(0)$$
$$g_{tool} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 & 0 \\ c_2 & -s_2 & 0 & 0 \\ -s_2 & -c_2 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & l_2 \\ c_3 & -s_3 & 0 & 0 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1.2 Solve the inverse kinematics problem using the Paden-Kahan subproblems

To solve the inverse kinematics problem we take two steps:

Step 1: finding θ_1 and θ_2 We have:

$$g_{tool} = e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_1}\theta_1}$$
$$e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_1}\theta_1} = g_d \; g_{tool}^{-1}(0) = g_1$$

Then we can select a point at the basis of the end effector p_w such that:

$$e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_2}\theta_2} p_w = g_1 \; p_w$$

And we fall into the category of **Paden-Kahn subproblem 2**, where $p = p_w$ and $q = g_1 p_w$:

$$e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_2}\theta_2} p = q$$

Which can be solved by:

$$e^{-\hat{\xi_1}\theta_1} q = c$$
$$e^{\hat{\xi_2}\theta_2} p = c$$

Which are two instances of **Paden-Kahn subproblem 1**.In these problems, c is obtained by $c = z + r$, where r is the point of intersection of the axis of θ_1 and θ_2 (*i.e*, $[0, 0, l_1]^T$), and for z:

$$z = \alpha\omega_1 + \beta\omega_2 + \gamma(\omega_1\omega_2)$$

Where:

$$\alpha = \frac{(\omega_1^T\omega_2)\omega_2^T u - \omega_1^T v}{(\omega_1^T\omega_2)^2 - 1}$$
$$\beta = \frac{(\omega_1^T\omega_2)\omega_1^T v - \omega_2^T u}{(\omega_1^T\omega_2)^2 - 1}$$
$$\gamma^2 = \frac{||u||^2 - \alpha^2 - \beta^2 - 2\alpha\beta\omega_1^T\omega_2}{||\omega_1\omega_2||^2}$$
$$u = p - r$$
$$v = q - r$$

We can see that we may have zero, one or two possible solutions for this equation depending on the square root of γ.
Step 2: finding θ_3 We can find θ_3 by directly solving a **Paden-Kahn subproblem 1**:

$$e^{\hat{\xi_3}\theta_3} = g_2$$

1.3 Derive the spatial and body Jacobians

The spatial jacobian is computed by determining the locations and directions of the joint twists as a function of the joint angles. We have:

$$\xi_1 = \begin{bmatrix} -w_1 q_1 \\ w_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \xi_2 = \begin{bmatrix} -w_2' q_1 \\ w_2' \end{bmatrix} = \begin{bmatrix} l_0 sin(\theta_1) \\ -l_0 cos(\theta_1) \\ 0 \\ -cos(\theta_1) \\ -sin(\theta_1) \\ 0 \end{bmatrix}$$

Also, we consider the velocity associated with point q_w at the wrist. The wrist is located at:

$$q_w' = \begin{bmatrix} 0 \\ 0 \\ l_0 \end{bmatrix} + e^{\hat{z}\theta_1} e^{-\hat{x}\theta_2} \begin{bmatrix} 0 \\ l_1 \\ 0 \end{bmatrix} = \begin{bmatrix} -l_1 sin(\theta_1) cos(\theta_2) \\ l_1 cos(\theta_1) sin(\theta_2) \\ l_0 - l_1 sin(\theta_2) \end{bmatrix}$$

And for the last rotation we have:

$$w_3' = e^{\hat{z}\theta_1} e^{-\hat{x}\theta_2} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -s_1 c_2 \\ c_1 c_2 \\ -s_2 \end{bmatrix}$$

Finally, we have the Jacobian:

$$J^s_{st} = \begin{bmatrix} 0 & -w'_2 q_1 & -w'_3 q'_w \\ w1 & w'_2 & w'_3 \end{bmatrix}$$

To obtain the body Jacobian, we simpply multiply the spatial jacobian by the adjoint transformation:

$$J^s_{st} = Ad_{g_{st}(\theta)} J^b_{st}(\theta)$$
$$J^b_{st} = [Ad_{g_{st}(\theta)}]^{-1} J^s_{st}(\theta)$$

2 Resisting Force at Singular Configuration

At a singular configuration, it is possible that an end-effector wrench lies in the null space of the Jacobian transpose. If this happens, these wrenches can be balanced if we apply any torque configuration, even zero torque at the motors.

The null space of the jacobian transpose can be understood physically as a direction in which we cannot move the robot given a particular joint configuration. For a simple example of two rotation joints in series with the same rotation axis in Z, the null space would correspond to movements that try to move the arm out of the XY plane, and also movements that try to bring the end effector in the direction of the center of the arm when both links are aligned. In this simple exemple it is easy to see that forces along the directions of the null space of the jacobian transpose would be balanced solely by the structure of the robot, and without any application of joint torques.

3 Iterative Inverse Kinematics Methods

3.1 Implement a function which computes a Jacobian matrix for the marker tip of the Barrett WAM TM arm

We implement the spatial jacobian using twist coordinates as described in equation 3.54 on page 116 of the MLS text book:

$$J^s_{st}(\theta) = [\xi_1 \xi'_2 ... \xi_n]$$
$$\xi'_i = Ad_{(e^{\hat{\xi_1}\theta_1} e^{\hat{\xi_2}\theta_2} .. e^{\hat{\xi_{i-1}}\theta_{i-1}})} \xi_i$$

We define a static reference frame which is Frame 0 as depicted in the Barret WAM kinematics' PDF. Then, we find a q and Ω corresponding to each joint, and compute the respective twists. These are:

$$q_1 = [0,0,0]^T; \quad \omega_1 = [0,0,1]^T; \quad \xi_1 = [\omega_1 \times q_1;\ \omega_1]^T$$
$$q_2 = [0,0,0]^T; \quad \omega_2 = [0,1,0]^T; \quad \xi_2 = [\omega_2 \times q_2;\ \omega_2]^T$$
$$q_3 = [0.045,0,0.550]^T; \quad \omega_3 = [0,0,1]^T; \quad \xi_3 = [\omega_3 \times q_3;\ \omega_3]^T$$
$$q_4 = [0.045,0,0.550]^T; \quad \omega_4 = [0,1,0]^T; \quad \xi_4 = [\omega_4 \times q_4;\ \omega_4]^T$$
$$q_5 = [0,0,0.850]^T; \quad \omega_5 = [0,0,1]^T; \quad \xi_5 = [\omega_5 \times q_5;\ \omega_5]^T$$
$$q_6 = [0,0,0.850]^T; \quad \omega_6 = [0,1,0]^T; \quad \xi_6 = [\omega_6 \times q_6;\ \omega_6]^T$$
$$q_7 = [0,0,0.910]^T; \quad \omega_7 = [0,0,1]^T; \quad \xi_7 = [\omega_7 \times q_7;\ \omega_7]^T$$

Then the manipulator jacobian is given by the following table:

Table 1: Manipulator Jacobian

0.0000	0.0000	0.0045	-0.4877	-0.0942	-0.4995	-0.4305
0.0000	0.0000	-0.0448	-0.2123	0.1977	-0.6498	0.4524
0.0000	0.0000	0.0000	0.1465	0.0203	0.2139	0.1209
0.0000	-0.0998	0.1977	-0.3836	0.5334	-0.6981	0.7100
0.0000	0.9950	0.0198	0.9216	0.1692	0.6414	0.5622
1.0000	0.0000	0.9801	0.0587	0.8288	0.3183	0.4242

Code is available in ques_3_1_and_3_2.m

3.2 Compute a velocity v, in the same representation as your Jacobian, which would move the marker tip from xs in the direction of xd

Desired velocity, V_θ, for each joint angle is given by $\left[-7.6040, -13.4819, 12.2446, 40.4735, 8.3570, -27.3546, -15.7753\right]$

We obtain the above by first converting x_s and x_d to twist coordinates : $\xi_s = [v_s; \omega_s]^T$ and $\xi_d = [v_d; \omega_d]^T$, respectively. Code for the same in available in get_twist_from_pose.m and get_twist_from_homo.m Let x_s^{XYZ} denote the current XYZ position of the end effector

Then the error in the end effector twist can be calculated by :

$$e'_{twist} = \xi_s - \xi_d = \begin{bmatrix} v_s \\ \omega_s \end{bmatrix} - \begin{bmatrix} v_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} e_{trans} \\ e_\omega \end{bmatrix}$$

$$e_{twist} = \begin{bmatrix} e_{trans} + (x_s^{XYZ} \times e_\omega) \\ e_\omega \end{bmatrix}$$

Now, the required velocity in joint space is given by :

$$V_\theta = pinv(J_{st}^s(\theta)) * e_{twist}$$

3.3 Implement the Jacobian pseudoinverse iterative method using your results from questions 1 and 2

$$\theta_d^1 = \left[2.8100, -25.0397, -0.0350, 58.6791, -1.6397, -34.8739, -4.1386\right]$$
$$\theta_d^2 = \left[-19.2077, -33.6034, 21.8836, 76.0001, 16.1109, -41.0810, -25.2461\right]$$

Code is available in ques_3_3.m, and the trajectory files are available in results_3_3_1_trajectory.txt and results_3_3_2_trajectory.txt.

At each step, we update the current joint angles by:

$$\Delta\theta = \eta * pinv(J_{st}^s(\theta)) * e_{twist}$$

where we set $\eta = 0.01$

We stop when L2 norm over error between the current and the desired end effector twists is less than 0.01. Note that error is computed as defined in question 3.2 :

$$||error_{twist}|| < 0.01$$

Following figures shows how the L2 norm of twist error changes over each iteration

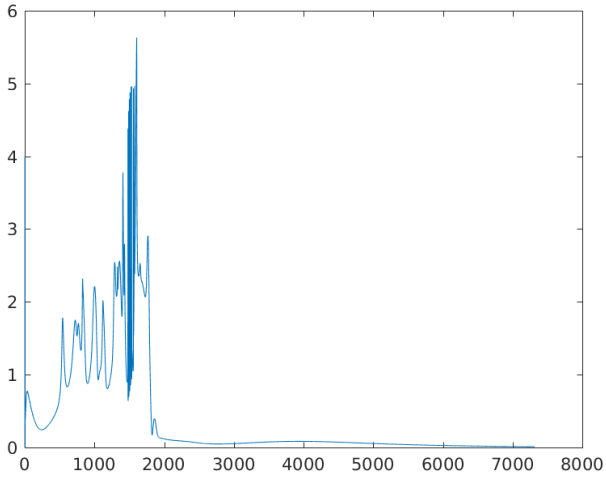

Figure 2: Error plot for x_{d1}. Y-axis shows norm of twist error, e_{twist} as defined in the text, X-axis denotes the iteration

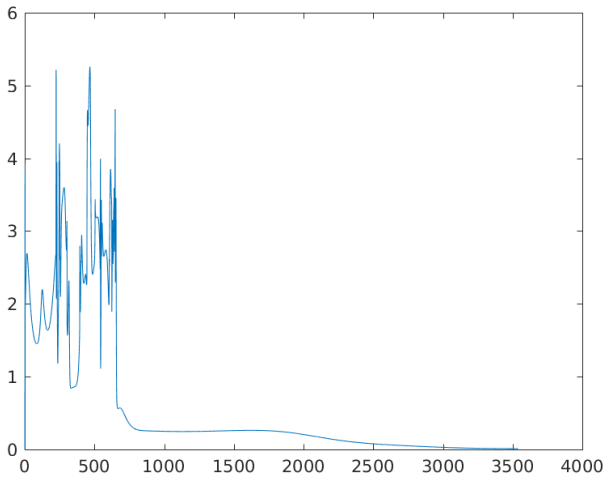

Figure 3: Error plot for x_{d2}. Y-axis shows norm of twist error, e_{twist} as defined in the text, X-axis denotes the iteration

We can see that we reach a fairly good solution by 2000 iterations, but due to our tight stopping criteria of L2 norm being less than 0.01, we take almost 7000 iterations.

3.4 Implement the damped least squares method

The hyperparams for DLS are identical to the pseudo inverse method, and we choose a regularizer cofficient, $\lambda = 0.001$

$$\theta_d^1 = [2.8045, -25.0378, -0.0264, 58.6815, -1.6260, -34.8700, -4.1265]$$
$$\theta_d^2 = [-38.1740, -102.6034, 41.1054, 220.7906, 30.1680, -116.0582, -48.4990]$$

Following figures shows how the L2 norm of twist error changes over each iteration

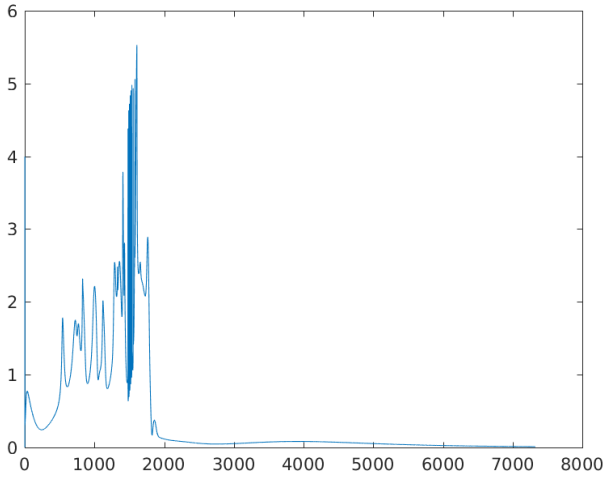

Figure 4: Error plot for x_{d1}. Y-axis shows norm of twist error, e_{twist} as defined in the text, X-axis denotes the iteration

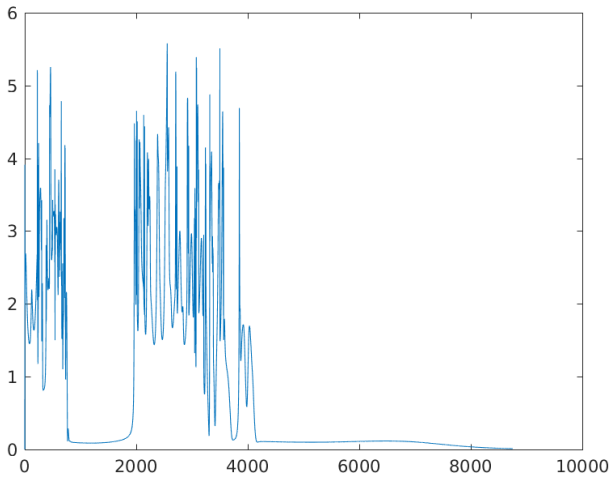

Figure 5: Error plot for x_{d2}. Y-axis shows norm of twist error, e_{twist} as defined in the text, X-axis denotes the iteration

Code is available in ques_3_4.m, and the trajectory files are available in results_3_4_1_trajectory.txt and results_3_4_2_trajectory.txt.

Similar to the previous part, we can see that we reach a fairly good solution by 2000 iterations, but due to our tight stopping criteria of L2 norm being less than 0.01, we can take over 7000 iterations.

For problem 1, DLS and pseudo inverse are almost identical and converge to the same joint configuration, unlike problem 2. The reason for this is that in problem 2, the desired end effector pose is more far away from the start pose, as compared to problem 1, and DLS differs as it tends to avoid singularities by using a regularizer in its cost function.

4 Virtual Model Control, Kineto-Statics Duality, and Parallel Chains

4.1 what are the initial ankle and knee joint angles?

After choosing the sensible values corresponding to the figure. The initial joint angles are:

$$\theta_{la} = -0.8462, \quad \theta_{lk} = 1.2025, \quad \theta_{ra} = -0.3563, \quad \theta_{rk} = 1.2025$$

4.2 Create a program which numerically integrates the simple dynamics of the trunk, given its initial state

The below graph shows that the states are eventually stablized.

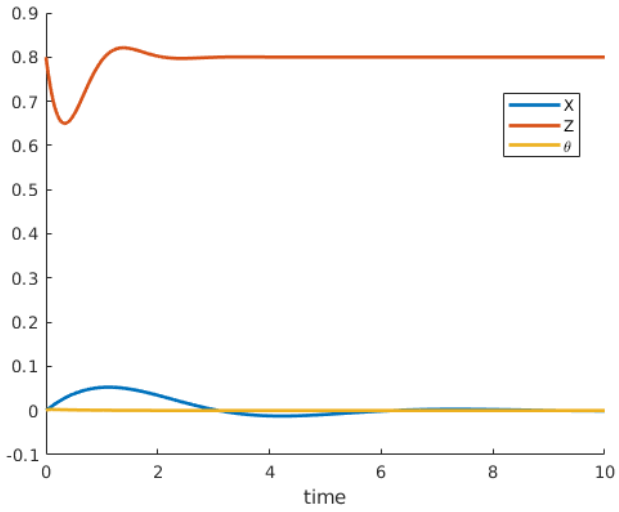

Figure 6: x, z, θ over time

4.3 Neglecting the motion of the legs, add to your program a part to compute estimated joint torques which generate the virtual forces during each time step

As we can see from the below graph, the left and right knee motors are close to saturation but still lower than 150 Nm.

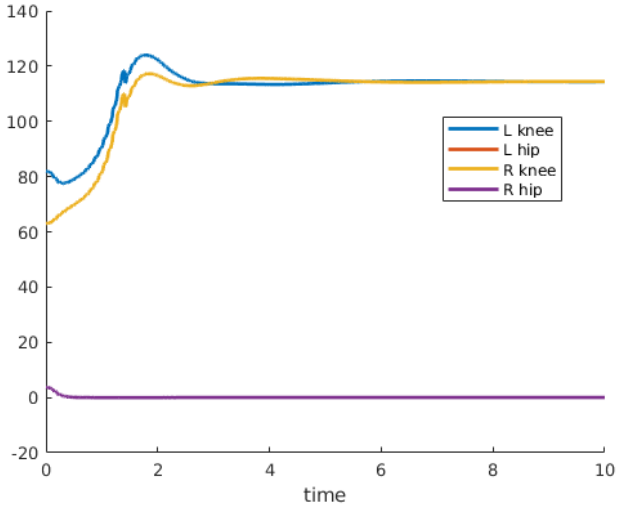

Figure 7: Joint torque over time

4.4 Repeat (c) but now use the IK to adjust the configuration of each leg