

Homework 3

Ting-Hao Liu

I. METHOD

A. Force/Energy Evaluation

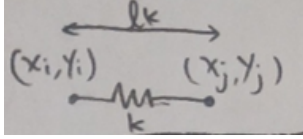


Figure 1: Shown is the spring connecting 2 points with stiffness k N/m and initial length l_k .

Spring energy:

$$E_s = \frac{1}{2} k l_k \varepsilon^2 \quad (1)$$

$$\varepsilon = \frac{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{l_k} \quad (2)$$

Spring force:

$$f_s = k l_k \varepsilon (\nabla \varepsilon) \quad (3)$$

Bending energy:

$$E_b = \frac{1}{2} EI \frac{\kappa^2}{l_k} \quad (4)$$

Where EI is bending stiffness, and κ is curvature.

Bending force:

$$f_b = \frac{EI}{l_k} \kappa (\nabla \kappa) \quad (5)$$

B. Time Stepping

Consider the inertia force:

$$f_{inertia} = \frac{m}{\Delta t} \left(\frac{x_{new} - x_{old}}{\Delta t} - u_{old} \right) \quad (6)$$

The Jacobian of inertia:

$$J_{inertia} = \frac{diag(m)}{(\Delta t)^2} \quad (7)$$

Where x is position vector, and u is velocity vector. m is the mass of nodes, and Δt is sampling time.

For each time stamp, the total force and Jacobian are the combination from inertia, spring, bending and external.

$$f = f_{inertia} + f_s + f_b - f_{ext} \quad (8)$$

$$J = J_{inertia} + J_s + J_b - J_{ext} \quad (9)$$

For each free nodes, we want to find the new position and velocity such that

$$f_{inertia} + f_s + f_b - f_{ext} = 0 \quad (10)$$

We can design a "Integrator" function by know previous position, velocity and the index of free degree of freedom (I_{free}) matching position and velocity vectors:

$$x_{new}, u_{new} = \text{Integrator}(x_{old}, u_{old}, I_{free}):$$

$$x_{new} = x_{old}$$

$$eps = 10^{-6}$$

$$err = 1$$

while $err > eps$:

Calculate force (f) and Jacobian (J) by x_{new} , x_{old} and u_{old} with Eq. 8, 9.

$$f_{free} = f[I_{free}]$$

$$J_{free} = J[I_{free}, I_{free}]$$

$$\Delta x_{free} = J_{free} \setminus f_{free}$$

$$x_{new}[I_{free}] = x_{old}[I_{free}] + \Delta x_{free}$$

$$err = \|f_{free}\|_2$$

$$u_{new} = \frac{x_{new} - x_{old}}{\Delta t}$$

For each iteration, we'll keep sending current position, velocity vectors into "Integrator" to update new state.

C. Dirichlet Constraints

Consider our elastic beam has N nodes, which means there are $2N$ elements in position and velocity vectors, and we can denote their index from 0 to $2N - 1$. In our problem formulation, we fix first, and final 2 nodes, so the index of free degree of freedom (I_{free}) is

$$I_{free} = [2 \ 3 \ \dots \ 2N - 5] \quad (11)$$

For each time step, we can implement the following pseudocode:

$t = 0: \Delta t: t_f$

For $k = 1: \text{len}(t) - 1$

$$x_{old}[2N - 2] = x_c[k]$$

$$x_{old}[2N - 1] = y_c[k]$$

$$x_{old}[2N - 4] = x_c[k] - \Delta L \cos \theta_c[k]$$

$$x_{old}[2N - 3] = y_c[k] - \Delta L \sin \theta_c[k]$$

$$x_{new}, u_{new} = \text{Integrator}(x_{old}, u_{old}, I_{free})$$

$$x_{old} = x_{new}$$

$$u_{old} = u_{new}$$

Where $x_c[k]$, $y_c[k]$, $\theta_c[k]$ are the control input at time step k .

D. Controller Design

At each time step k , we use PI controller based on the reference on x and y axis ($r_x[k]$ and $r_y[k]$). The following is the pseudocode:

$t = 0: \Delta t: t_f$

Number of middle node $N_{middle} = N // 2 + 1$

$$\text{Error in x-axis: } e_x = r_x[0] - x_{old}[2N_{middle} - 2]$$

$$\text{Error in y-axis: } e_y = r_y[0] - x_{old}[2N_{middle} - 1]$$

$$\text{Accumulated error in x-axis: } e_{x,acc} = e_x \Delta t$$

Accumulated error in y-axis: $e_{y,acc} = e_y \Delta t$

For $k = 1: \text{len}(t) - 1$

$$x_c[k] = x_c[k-1] + k_{p,x}e_x + k_{i,x}e_{x,acc}$$

$$y_c[k] = y_c[k-1] + k_{p,y}e_y + k_{i,y}e_{y,acc}$$

$$\theta_c[k]$$

$$= \text{atan2}(x_{old}[2N-1]$$

$$- x_{old}[2(N_{middle} + 1) - 1], x_{old}[2N-2]$$

$$- x_{old}[2(N_{middle} + 1) - 2])$$

After Dirichlet constraints are implemented and new position is obtained, the errors can be updated:

$$e_x = r_x[k] - x_{new}[2N_{middle} - 2]$$

$$e_y = r_y[k] - x_{new}[2N_{middle} - 1]$$

$$e_{x,acc} += e_x$$

$$e_{y,acc} += e_y$$

II. SIMULATION RESULTS

A. Control Inputs over Time

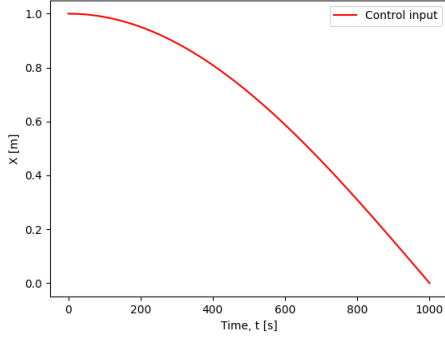


Figure 2: Shown is control input in x-axis (x_c).

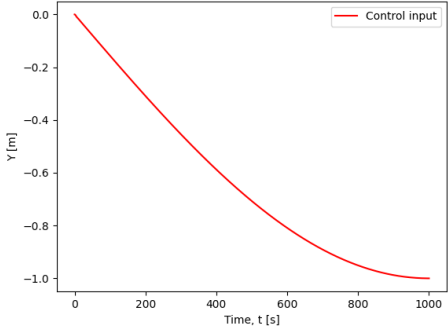


Figure 3: Shown is control input in y-axis (y_c).

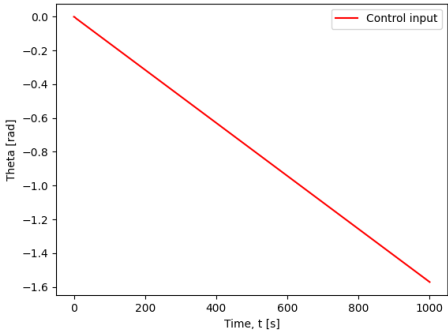


Figure 4: Shown is angle control input (θ_c).

B. Tracking Results

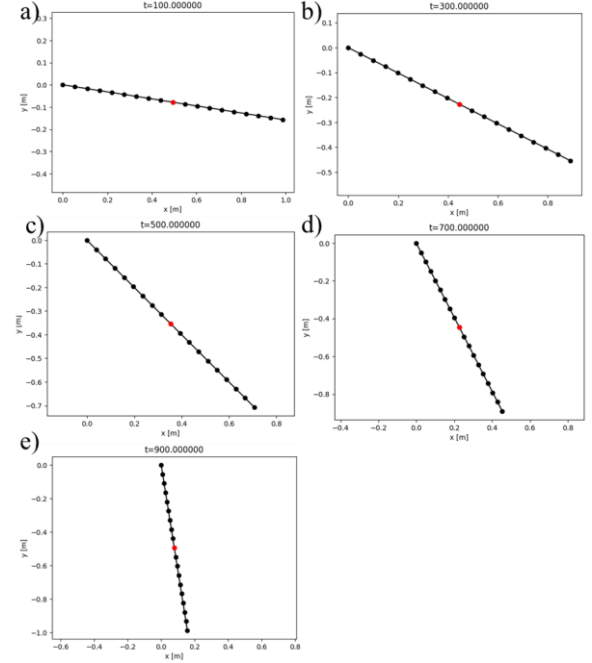


Figure 5: Shown are 5 different configurations during trajectory tracking.

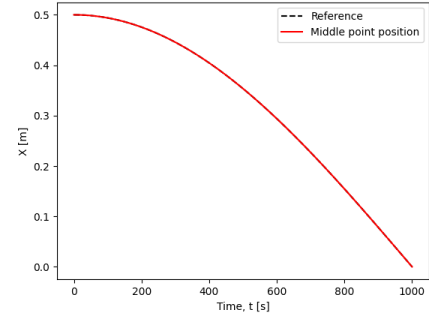


Figure 6: Shown is the middle point tracking result in x-axis.

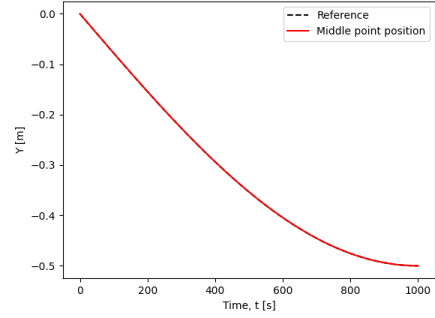


Figure 7: Shown is the middle point tracking result in y-axis.

III. DISCUSSION

If any control input ($x_c[k]$, $y_c[k]$, or $\theta_c[k]$) is out of robot limits, we can set them to the robot limit (saturation).