ME455 Active Learning HW4

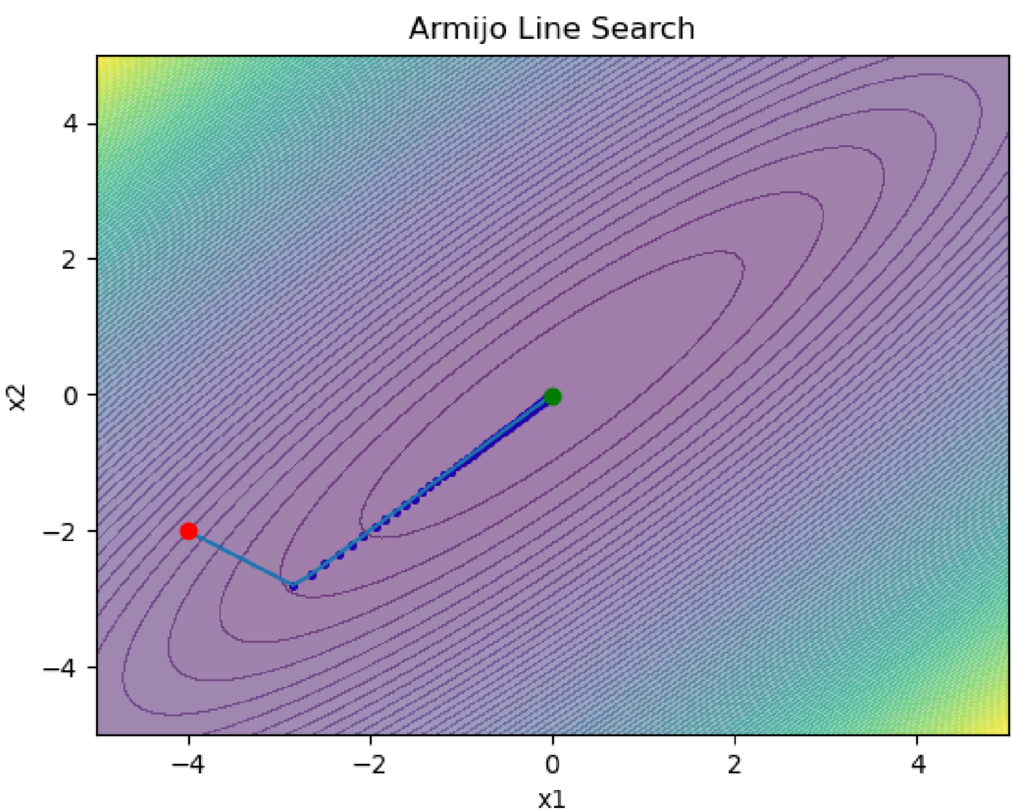
Graham Clifford

5/9/2024

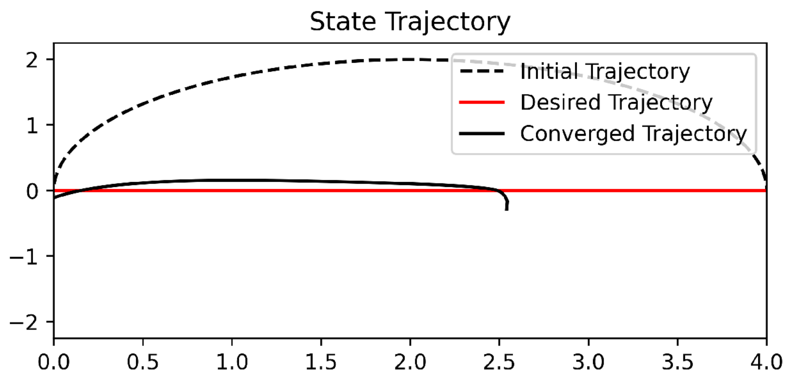
Problem 1

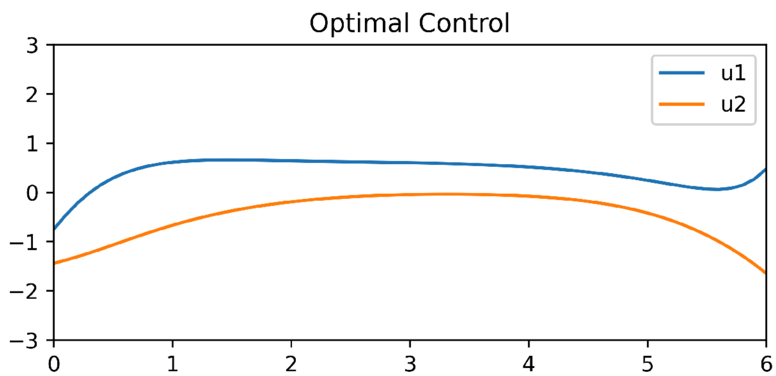
Please see handwritten pages for problem 1.

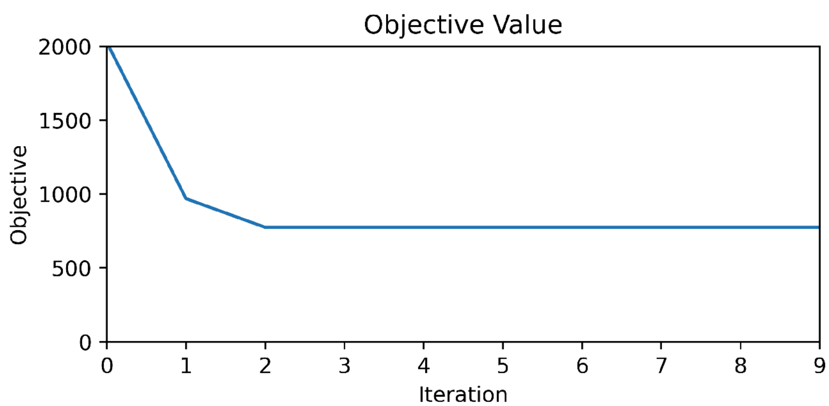
Problem 2



Problem 3







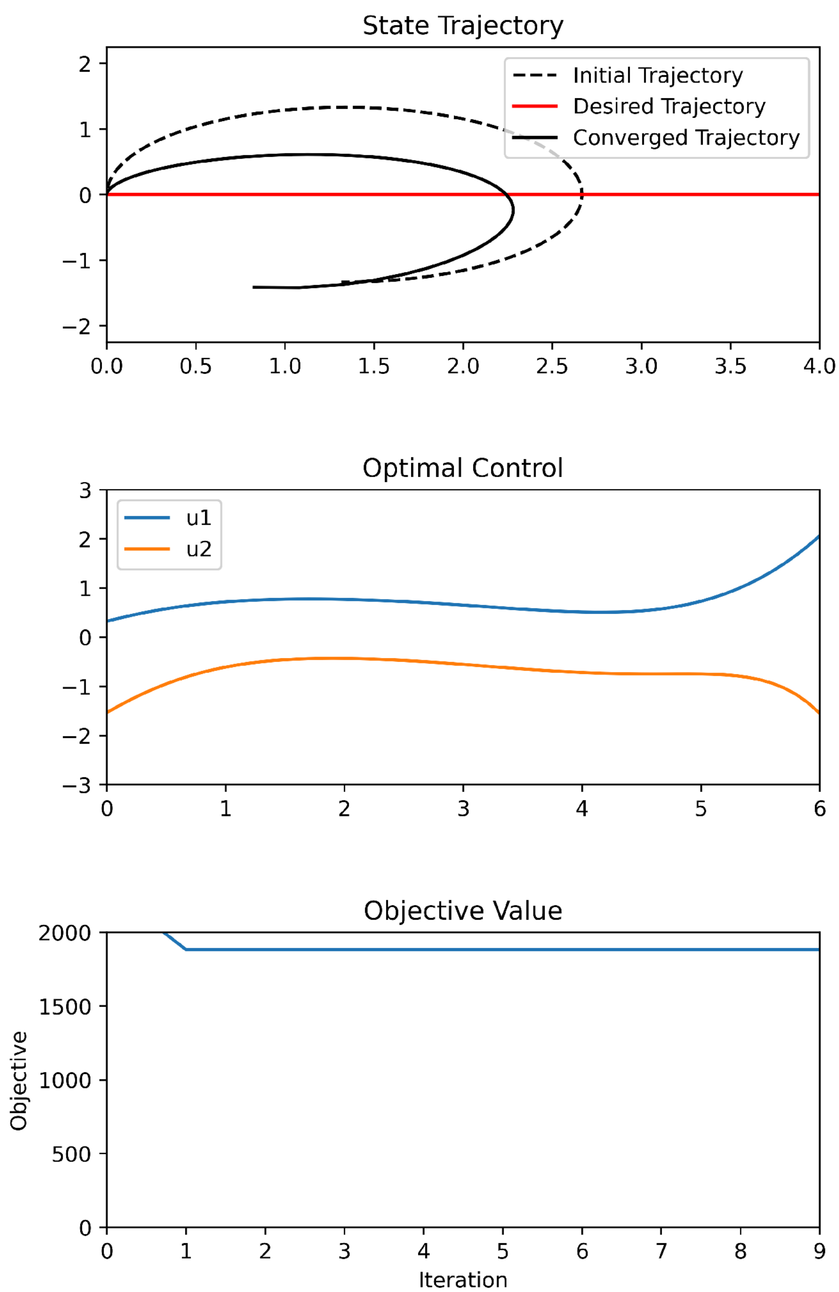
I'm not really sure why I couldn't get my plots to look like the example plots, however I think there must be something wrong with either my iLQR algorithm or the way I calculate my loss. I'm doing my iLQR algorithm where I calculate the cumulative loss for the entire trajectory with the current control trajectory, as well as the cumulative loss for the entire trajectory with the modified control sequence (u + alpha * gamma). Maybe this is wrong, but it seemed to me like the example code was encouraging this strategy. This leaves me with a loss that plateaus at 1000, which is kinda weird.

The iteration above was with all default parameters. Here's an iteration where the only thing I changed was the original control trajectory: dt = 0.1 x0 = np.array([0.0, 0.0, np.pi/2.0]) tsteps = 63

init_u_traj = np.tile(np.array([1.0, -0.75]), reps=(tsteps,1))

Q_x = np.diag([10.0, 10.0, 2.0]) R_u = np.diag([4.0, 2.0]) P1 = np.diag([20.0, 20.0, 5.0])

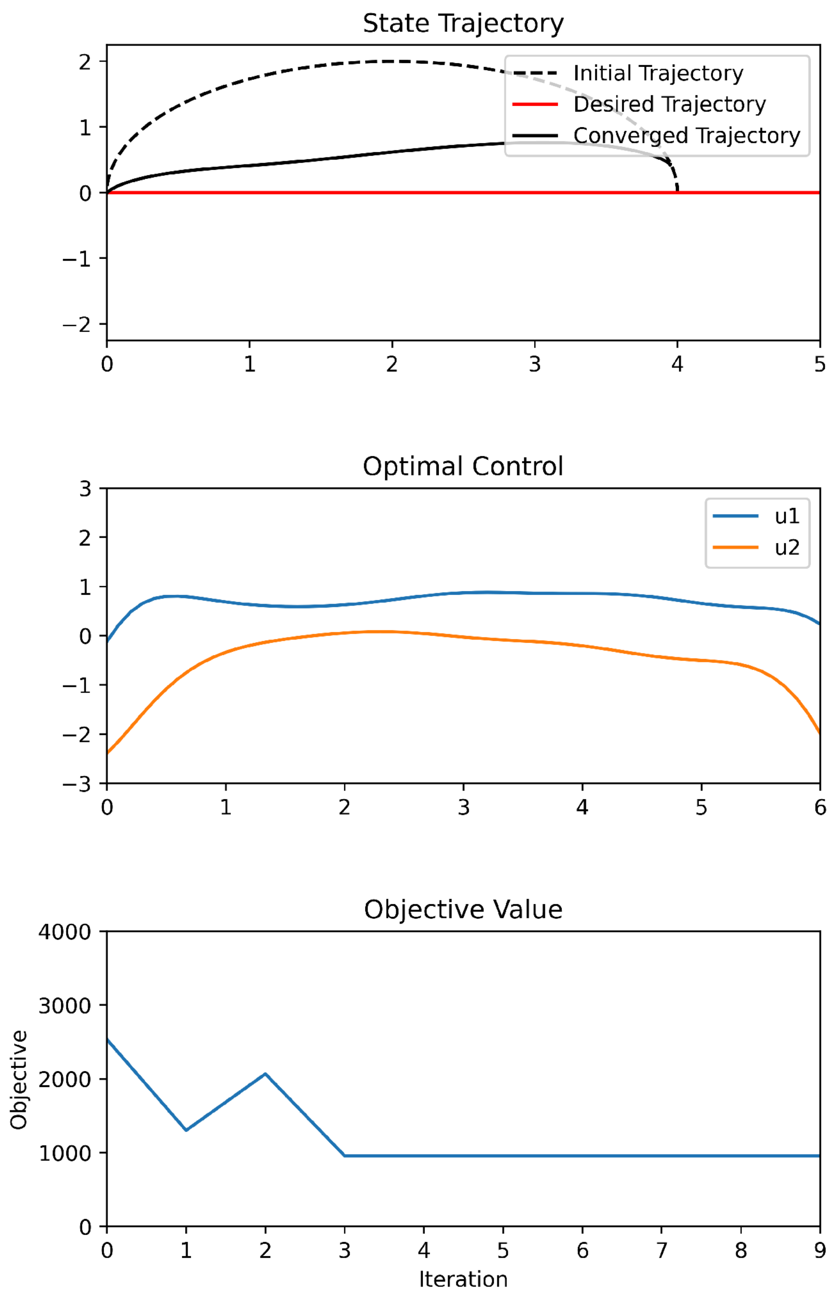
Q_z = np.diag([5.0, 5.0, 1.0]) R_v = np.diag([2.0, 1.0])



The converged trajectory seems to be focusing mainly on the initial trajectory, and not the desired trajectory.

I switched the initial control trajectory back to the default, and then tried to get the final position of the differential drive robot to be close to the final position of the initial trajectory.

Turns out all I really needed to do was change one of the values on the diagonal in the Qx matrix: Q_x = np.diag([95.0, 10.0, 2.0])



This algorithm was tough to handle, I think it’s possible it’s slightly working, however I can’t be sure.

ME455 HW4 Graham Clifford

Problem #1:

$$v(t)^{[k]} = \underset{v(t)}{\arg\min} \int_0^T D_1 \ell(x(t)^{[k]}, u(t)^{[k]})\, z(t) + D_2 \ell(x(t)^{[k]}, u(t)^{[k]}) \cdot v(t)\, dt$$

$$+ D_x m(x(T)^{[k]}) \cdot z(T)$$

$$+ \int_0^T z(t)^T Q_z z(t) + v(t)^T R_v v(t)\, dt$$

$$= \int_0^T D_1 \ell(x(t)^{[k]}, u(t)^{[k]}) z(t) + z(t)^T Q_z z(t) + D_2 \ell(x(t)^{[k]}, u(t)^{[k]}) v(t) + v(t)^T R_v v(t)\, dt$$

$$+ D_m(x(T)^{[k]}) \cdot z(T)$$

$$= \int_0^T z(t) \underbrace{\left[D_1 \ell(x(t)^{[k]}, u(t)^{[k]}) + z(t)^T Q_z\right]}_{a_z(t)^T} + v(t) \underbrace{\left[D_2 \ell(x(t)^{[k]}, u(t)^{[k]}) + v(t)^T R_v\right]}_{b_v(t)^T} dt$$

$$+ D_m(x(T)^{[k]}) \cdot z(T)$$

now we can sub in $a_z(t)$ & $b_v(t)$...

$$0 = \rho(t)^T B(t) + v(t)^T R_v + D_2(\ell(x(t)^{[k]}, u(t)^{[k]}))$$

solve for $v(t)$

$$v(t) = -R_v^{-1}\left[B(t)^T \rho(t) + D_2(\ell(x(t), u(t)))\right]$$

and sub in $v(t)$

$$\dot{z}(t) = A(t) z(t) - B(t) R^{-1}\left[B(t)^T \rho(t) + D_2(\ell(x(t)^{[k]}, u(t)^{[k]}))\right]$$

$$\dot{\rho}(t) = -A(t)^T \rho(t) - Q_z z(t) - D_1(\ell(x(t)^{[k]}, u(t)^{[k]}))$$

$$\boxed{\begin{bmatrix} \dot{z} \\ \dot{\rho} \end{bmatrix} = \begin{bmatrix} A(t) & -B(t) R^{-1} B(t)^T \\ -Q_z & -A(t)^T \end{bmatrix} \begin{bmatrix} z \\ \rho \end{bmatrix} + \begin{bmatrix} -B(t) R^{-1} D_2(\ell(x(t)^{[k]}, u(t)^{[k]})) \\ -D_1(\ell(x(t)^{[k]}, u(t)^{[k]})) \end{bmatrix}}$$