Model Predictive Control - EL2700

Assignment 2: Finite-time Optimal Control

Group 7: Christopher Biel, Davide Torrini

1 System Model: Q1 & Q2

Implementing A_c and B_c in `Astrobee.cartesian_ground_dynamics()` as:

$$A_c = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B_c = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_Z} \end{bmatrix}$$

allows the execution of `casadi_c2d`, which return the values for A_d and B_d. Using the parameters $\delta t = 0.1$ s, $m = 20.9$ kg and $I_Z = 0.25$ kgm^2 this results in the discrete system matrices A_d and B_d:

$$A_d = \begin{bmatrix} 1 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad B_d = \begin{bmatrix} 0.00024 & 0 & 0 \\ 0 & 0.00024 & 0 \\ 0.00478 & 0 & 0 \\ 0 & 0.00478 & 0 \\ 0 & 0 & 0.02 \\ 0 & 0 & 0.4 \end{bmatrix}$$

2 Finite-time Optimal Control: Q3& Q4

After implementation of the rendezvous constraints in `FiniteOptimization.__init__()` for the 2D case, the optimization and simulation produces the result shown in fig. 1. In fig. 2 we can see that Bumble loops behind Honey to approach Honey on its position on the circle at time $t = 25$ s.

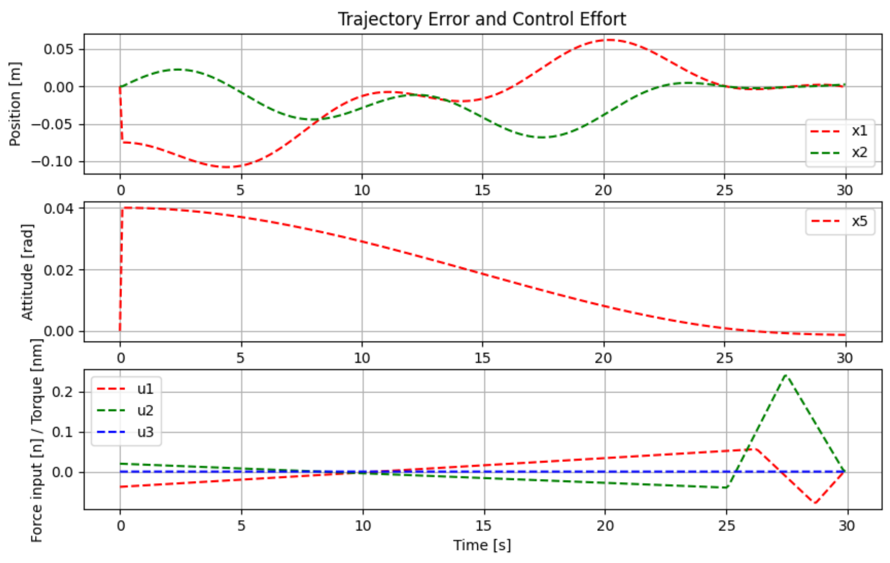

Figure 1: The trajectory error converges close to zero (within the tolerance band) for the interval $t = [25, 30]$

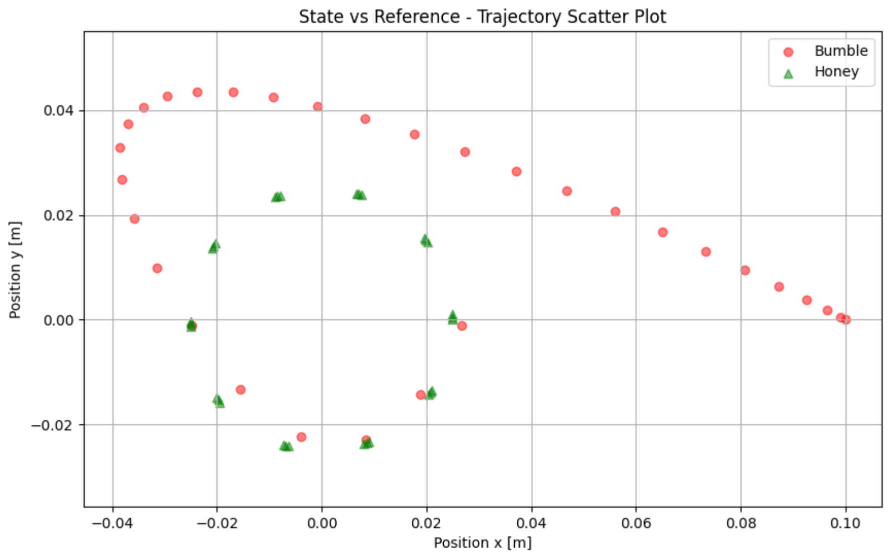

Figure 2: The trajectory scatter visualizes the approach of Bumble to match Honey. The points are scattered at a regular interval of 1 s.

3 Broken Thruster: Q5 & Q6

We can observe (fig. 3) that the trajectory greatly deviates from the expected optimal trajectory as computed by the optimization. We do not rendezvous with bumble, but instead have a trajectory error (which varies for multiple simulations, based on the random disturbance in the actuator). This is due to a lack of loop-closure: The optimal trajectory only works for the given x_0 and perfectly working thrusters. With a thruster defect this changes the state dynamics of the system. Since there is no feedback of the current state the control inputs do not compensate for the growing deviation from the target trajectory. Bumble ends up in a different location than expected (as can be seen in 4). We can allow for loop-closure in multiple ways:

- Re-Compute the optimal trajectory for every time step. This allows the current state to influence the control inputs as the new x_0 for every optimization. This is however computationally inefficient and does not address the deviation of the expected behaviour from the real behaviour (This could mean, that the optimization never reaches its goal, since it calculates with wrong model dynamics)
- Implement a LQR design which computes a control matrix L for the system input $u = -L * x$ and thus directly introduces state feedback. This is computationally more efficient but also does not consider the changing model dynamics. By adding an integral error term we can prevent constant control errors.

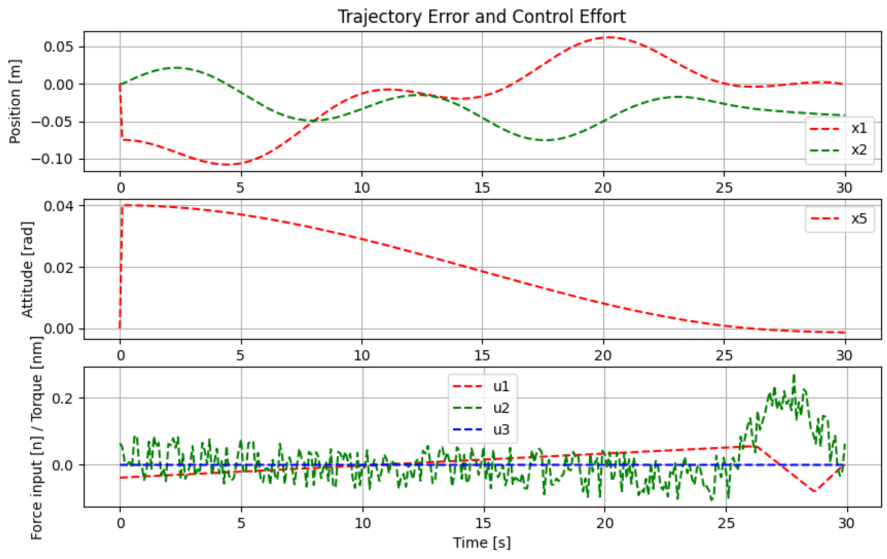

Figure 3: Simulating the optimal trajectory with actuator disturbance results in a large error in the trajectory.

4 Extra Dimensions: Q7

The system model for 3D translation and 1D rotation with state $x = [p_X, p_Y, p_Z, v_X, v_Y, v_Z, \Theta, \omega]^T$ and input $u = [f_X, f_Y, f_Z, \tau_Z]^T$ is $\dot{x} = A_c * x + B_c * u$ with:

$$A_c = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B_c = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{I_Z} \end{bmatrix}$$

After implementation in `Astrobee.cartesian_3d_dynamics()` the simulation can be run and results in fig. 5 and fig. 6.

Since Honey now moves in 3D Bumble also has to adapt its path to approach with a matching trajectory. The addition of two states increases the complexity of the problem significantly, the solver now calculates ≈ 9 times longer than for the 2D case (the optimization time varies between runs). The final cost is also higher, which can mainly be attributed to the higher cost of the actions performed to rendezvous in 3D.

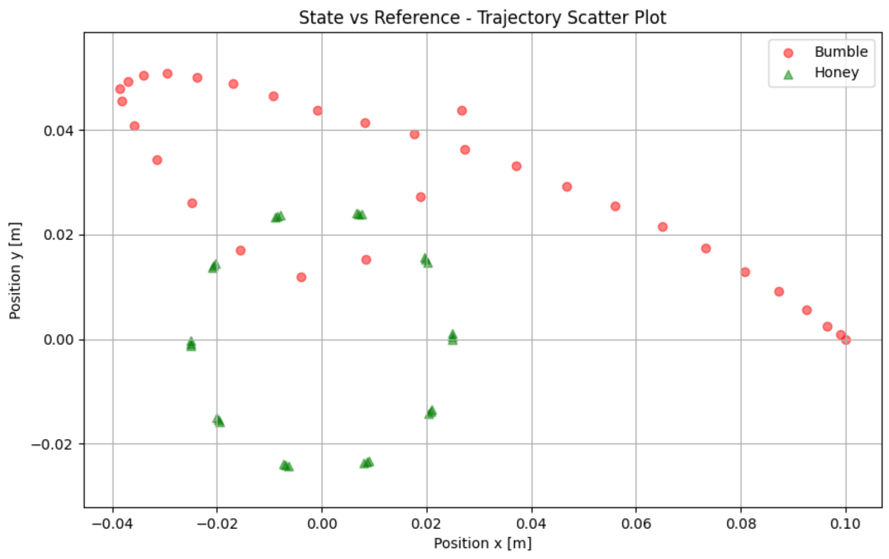

Figure 4: Bumble ends up in a different location, trying to rendezvous with Honey but not reacting to the changing system dynamics.

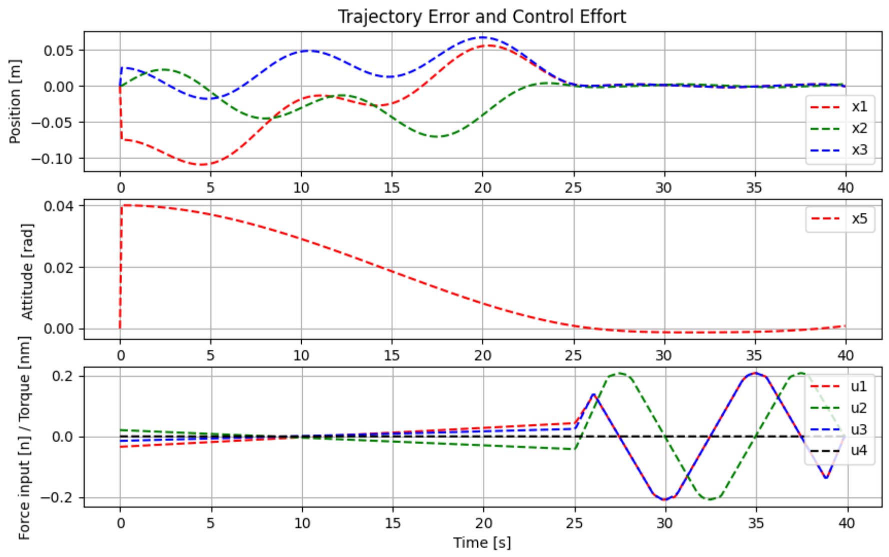

Figure 5: Trajectory error for the 3D simulation. Bumble follows the circular movement of Honey from 25 s to 40 s as can be seen by the sinusoidal control input.

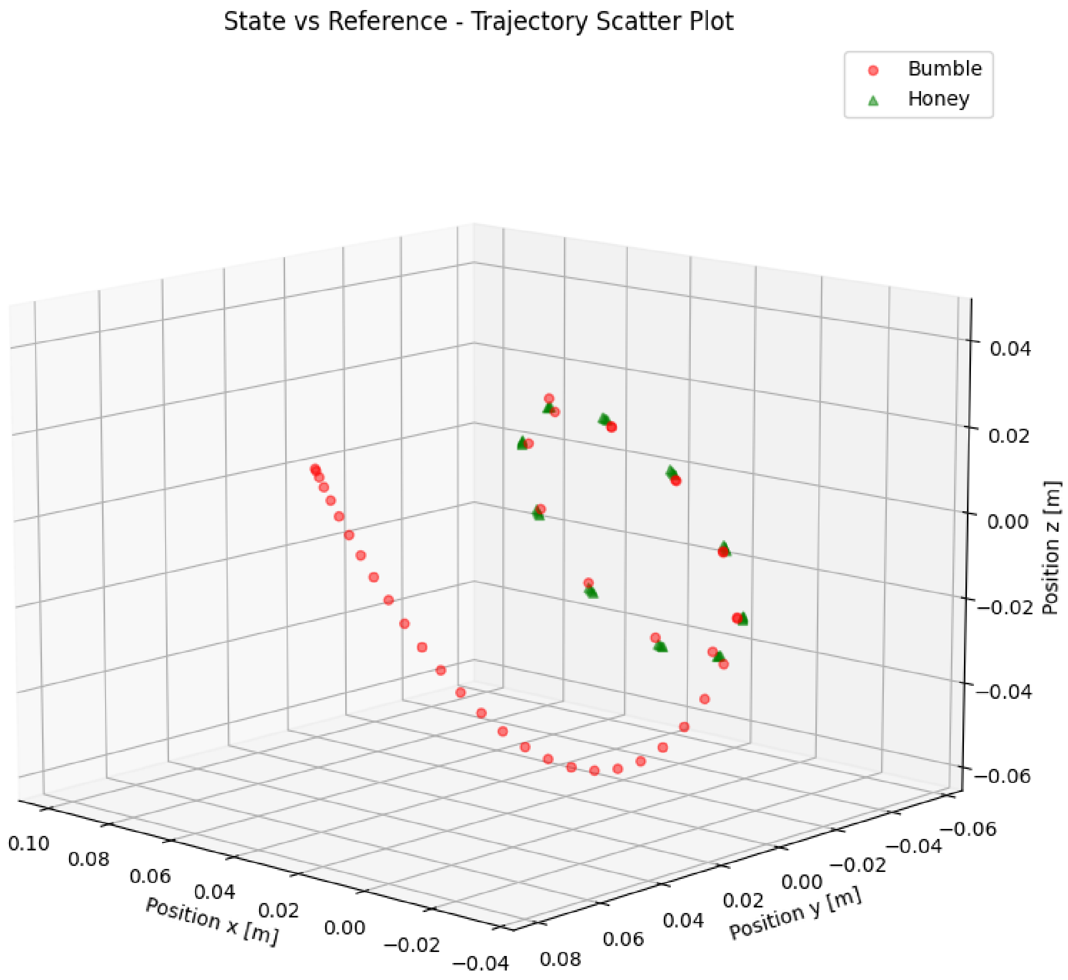

Figure 6: 3D Visualisation of the resulting trajectory shows how Bumble approaches Honey. Bumble starts of with a movement in negative z- and positive x-&y-direction. It loops under and around the circulating Honey to approach from negative x-direction to match the trajectory.