16-665: Robot Mobility
Fall, 2018

Air Mobility
Project Description
Due: October 30, 2018

Overview and Objectives

The *Air Mobility* project emphasizes the generation of optimal time-parameterized piecewise continuous trajectories and feedback control design to enable an aerial robot (in simulation) to fly along a pre-defined path. The project seeks to:

- Develop a basic state machine to facilitate simulation that enables the robot to takeoff, hover, track a trajectory, and land;
- Introduce time-parameterized trajectories given fixed initial and final endpoint constraints and bounded velocity and acceleration;
- Extend the formulation to include piecewise continuous trajectories with fixed initial and final endpoint constraints;
- Evaluate tracking performance given different levels of flight performance and trajectory design (from slow to fast, straight and curved paths).

Project Report Format and Grading Criteria

Each individual is expected to submit a project report that will be graded out of a total of 100 points based on content correctness (40%), quality (20%), completeness (20%), and clarity of presentation (20%). Individual point values are indicated next to each question in square brackets (e.g., [1] indicates a value of 1 point). Consider the following *representative* questions when preparing the report.

- Correctness: Is there a notable or unexplained error or phenomena presented in the results not addressed by the discussion? Do the results correspond to the project topic and adequately support the discussion claims? Is there sufficient statistical evidence to show that the results are repeatable and valid?
- Quality: Do the results present an acceptable level of performance (e.g., error) and is there a satisfactory explanation as to why the performance is acceptable given the system characteristics and methodology?
- Completeness: Do the results and discussion adequately address the full problem with consideration of the multiple problem facets? Are there any notable omissions in the approach, discussion, or results presentation? Did you provide the Matlab code in the report?
- Clarity of presentation: Is the discussion cohesive, coherent, devoid of grammatical errors, and self-contained? Do the figures and tables adequately present the data? Are the figures labeled and easily read when printed with appropriate color (or grayscale) schemes, line thickness, and structural formatting?

Collaboration: The project is expected to be completed by individuals (and is not a team project). Collaboration with one other person is acceptable. Collaborators must be acknowledged explicitly on the report by name. Failure to report collaborators or collaborate with more than one person will be viewed as a violation of the CMU Academic Integrity Policy (as detailed in the course syllabus).

Project Report Content

1. *System Diagram [5]:* Design a software pipeline that consists of multiple Matlab components that achieve the necessary structure and capabilities required to enable the simulation of a quadcopter platform. Provide a system diagram that includes the input and output associated with each software module and data content. Ensure that the diagram clearly indicates how information (data) flows through the software pipeline to be eventually applied to the simulated dynamic model. Similarly, the diagram should indicate how the simulated system state becomes available for closed-loop control after passing through multiple code modules.

2. *Hover Performance [10]:* Using the linearized feedback control policy developed in class, implement a PD feedback controller to enable the robot to hover at a desired location (e.g., z = 0.5 m). Create a simulation scenario where the robot transitions between multiple waypoints along the x-axis by specifying goals that increment by 10 cm in the x direction. Plot the error between the desired pose (position and orientation) and the actual pose. Indicate when new waypoints are sent to the system. Examine the convergence of the robot to each waypoint. Does the system oscillate about the waypoint? What happens when modifying the gains associated

with the position control (outer loop) and attitude control (inner loop)? Plot the response for multiple gains and discuss qualitatively the change in performance.

3. *Line-tracking Performance [10]:* Develop a PD line tracking controller to enable the robot to take-off from a starting location, go to a fixed height of 1.0 m, and return to the ground. Many approaches could be used (select your preferred approach). The most naive solution would follow a method similar to the approach detailed in (2) but reduce the magnitude of the incremental change in the waypoint by a smaller value (e.g., 1 cm) and assign a desired velocity to each waypoint following a velocity profile (consider a ramp profile due to a constant positive, zero, then negative acceleration). Plot the error between the desired pose (position and orientation) and the actual pose. Examine the convergence of the robot to each waypoint. Does the system oscillate about the waypoints? What happens when modifying the gains associated with the position control (outer loop)? Plot the response for multiple gains and discuss qualitatively the change in performance.

4. *State Machine [5]:* Develop and detail a state machine that enables the platform to takeoff to a pre-specified height, hover, track trajectories, and land. Provide a description of this state machine with an associated state machine diagram. One possible strategy is to develop a finite sequence of modes that transition based on the current and desired states as well as the prior mode. Consider the following possible approach:
    - The system begins in an **idle** state generating no (or null) control inputs.
    - A simulation trial begins by transitioning into a **takeoff** state that consists of a trajectory tracking controller from the current robot state to a desired hover state (fixed pose). When the robot approaches the desired hover state (based on the error between current and desired pose), the robot transitions into a **hover** mode.
    - The platform remains in **hover** mode for a small amount of time (e.g., 5 s) before transitioning into a **tracking** mode.
    - The platform tracks a specified trajectory and upon completion transitions into the **hover** mode.
    - After remaining in **hover** mode for a brief period of time, the robot transitions into **land** mode and begins a descent to the ground.

    Remark: The above state machine is a prerequisite for the remainder of the exercises in this project and provides a simple strategy for trial initialization and finalization. The following exercises will leverage this capability and assume that the system is operating in the **tracking** mode. Any requested error plots should apply only to this mode (and not include the takeoff, hover, and landing phases) unless explicitly requested.

5. *Gain Selection and Tuning [10]:* After commanding the robot to takeoff and hover, generate a command that tracks a single waypoint at [0, 0, 0.1] m (zero velocity). Plot the error response of the system with respect to the desired position, orientation, and linear and angular velocities. What are the rise (90%) and settling (10%) times associated with the position and linear velocities? What is the steady-state value? What is the maximum percent overshoot? Now provide a waypoint at the same position but with heading of 15 deg. What are the similar time domain performance characteristics of the heading controller? How do these values (position and heading) change given different gain values?

    Remark: Ensure that the values remain realistic for the actual robots (assuming a thrust-to-mass ratio of 2.6).

6. *LQR Controller Design and Evaluation [10]:* Repeat the evaluation in (2), (3), and (5) using an LQR-based feedback controller. How does the performance differ in terms of error response characteristics?

7. *Bounded Acceleration Trajectory Generation [10]:* Develop a straight line time-parameterized polynomial trajectory with initial and final endpoint constraints ([0, 0, 1] and [0, 0, 10], respectively; higher-order terms zero). Choose a time-scaling that ensures a bounded (maximum/minimum) acceleration less than 3 m/s^2. Generate a set of error plots that depict the performance of the platform when tracking the trajectory (using PD control) including the error in the pose and linear/angular velocities. Does the robot track the trajectory as expected? Is there any substantial error arising while tracking the trajectory? If so, what is the source? Continue to increase the desired acceleration bound. At what value does the system begin to exhibit degraded tracking accuracy? How does this empirical observation relate to the motor response? How does the performance improve if you artificially increase the motor gain configuration parameter thereby effectively upgrading the robot motors?

    Remark: Ensure that throughout these trials the maximum thrust stays within a reasonable level appropriate for the real-robot system. Remember to restore the motor model configuration value to the original value upon completion of this exercise.

8. *Minimum Energy Elliptical Trajectory [10]:* Define a piecewise continuous trajectory consisting of four waypoints: [0, 0, 1, 0], [2, 1, 1, 0], [0, 2, 1, 0], [-2, 1, 1, 0] (position, heading). To do so, generate an optimal (minimum energy) trajectory that visits the four waypoints (returning to the first; forming an ellipse) with an initial velocity of zero and an end velocity of 1 m/s at the first waypoint. The robot will start at rest, track the elliptical trajectory, and arrive at the starting location at a non-zero velocity (1 m/s). Generate a second trajectory phase given the same waypoints that starts with a 1 m/s velocity tangent to the ellipse and similar velocities at the other waypoints (all tangent to the ellipse in the direction of motion). To solve for the optimal trajectory, formulate the problem as a Quadratic Project (QP) and solve for the appropriate polynomial coefficients. Generate error plots that depict the tracking performance (using PD control). Generate a *cumulative error distribution* plot. How does the tracking performance change given differing velocity profiles?

   Remark: In order to ensure sufficient analysis for evaluation, track the second trajectory multiple times (repeating loops). To stop (prior to landing), introduce a third trajectory phase that slows the robot to stop (inverting the velocity specification for the first trajectory phase).

9. *Robot Pirouette [10]:* Repeat the same steps as specified in (7) but now with heading commands that cause the robot to point to the center of the ellipse (second trajectory phase). How does the performance change when introducing time-varying heading commands at high-speeds?

10. *Enforcing Smooth State Machine Transitions [10]:* The naive state machine proposed in (4) does not enforce the requirement that feedback control references can exhibit non-smooth (and non-trivial) jumps in the desired pose, velocities, and higher-order terms. As an example, consider the transition from the trajectory mode to the hover mode based on an error term that only considers the desired and current pose (ignoring the non-zero velocity and acceleration). Propose, implement, and evaluate two strategies to address and mitigate rapid changes in modes by saturation or gain selection. Possible strategies may include bounding changes in the input reference via a saturation function or choosing softer gains.

11. *Free Skate [10]:* Generate an aggressive maneuver (of your choosing) and evaluate the performance (following the analysis techniques detailed above). Ensure dynamic feasibility. Aggressive maneuvers invoke extreme roll/pitch attitudes (e.g., flips). What is the consequence of not modeling drag in this case? What angular velocities are achieved during the maneuver? Many IMUs are designed with a saturation of 300 deg/s. Pursue a maneuver that would approach the capability limits of a robot equipped with an IMU with a 300 deg/s saturation bound.

    Remark: The goal of this exercise is to exploit the concepts and implementations developed through the previous exercises to achieve high-performance flight. The only requirement here is that the result is interesting (and not slow/uninteresting). Flips, flight through windows, emulated perching are all possibilities but not required. Additional discussion, reading, and software modifications may be necessary to achieve these latter goals.