Test Bed for Analyzing Unmanned Aerial Vehicle Flight Safety

Danny Myers
Mentor: Gautam Biswas

ISIS
Tel (615) 343-7472 Fax (615) 343-7440
1025 16th Avenue South Nashville, TN 37212
www.isis.vanderbilt.edu

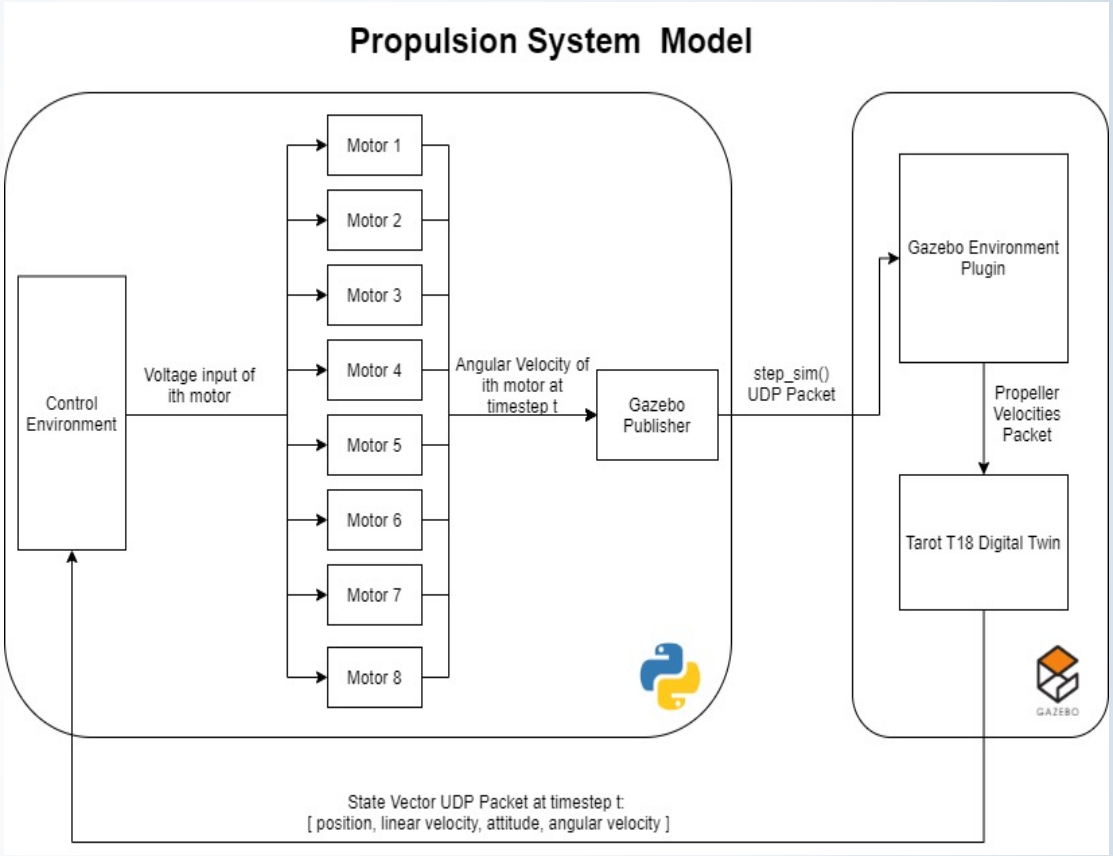
VANDERBILT UNIVERSITY

Project Purpose

- Motivation
  - Recent years have seen a proliferation of potential applications for unmanned aerial vehicles (UAV's) in urban environments (such as air taxis, package delivery, emergency transport to hospitals, etc.).
  - Using UAV's for these applications presents hazards in the event of vehicle degradation and faults. UAV battery life and motor efficiency can degrade from constant use. A motor fault can cause damage to people or property if a drone crashes in an urban environment.
- Project Goal
  - **System-Wide Safety Initiative (NASA)** : Use Reinforcement learning to develop fault-tolerant flight controllers that can detect vehicle degradation mid-mission and alter flight paths to maximize safety and minimize damage to both the vehicle and its surroundings.
  - **My project**: Create a simulation testbed in order to test re-planning algorithms and fault-tolerant control mechanisms.
  - Design and implement a "digital twin" of an existing octocopter drone (Tarot T18) to accurately train Reinforcement Learning controllers without the risks and time consumption of real-life flights and episodes, with the goal of transferring the trained flight controller to the physical Tarot T18 to fly in real-life missions.

Testbed Framework

- Gazebo
  - An open source robotics simulator capable of integrating programming with a customized environment.
  - Testbed environment contains the Tarot T18 digital twin and allows communication between the Gazebo model and the python model.
- Python Model
  - The Tarot T18 is modeled in python code which calculates the voltage being applied to the motors at each timestep.
  - Corresponding propeller rotation speed is calculated and communicated to Gazebo along with the command to step the simulation.



The python model steps the Gazebo simulation, ensuring synchronicity between environments.

Simulated Octocopter Trajectories

Trajectory 1: Nominal Flight

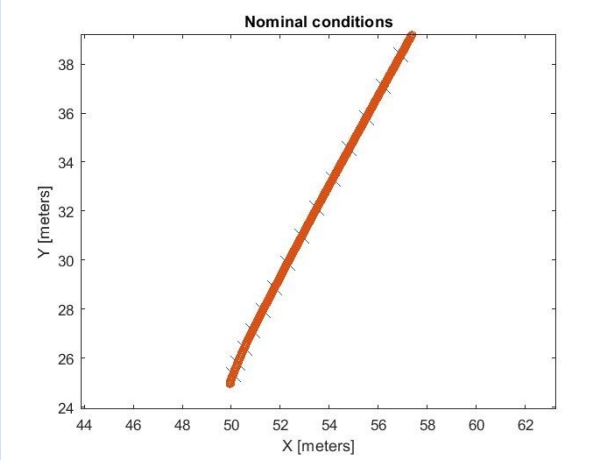
Simulated Octocopter Trajectories

Trajectory 2 : Motor Fault

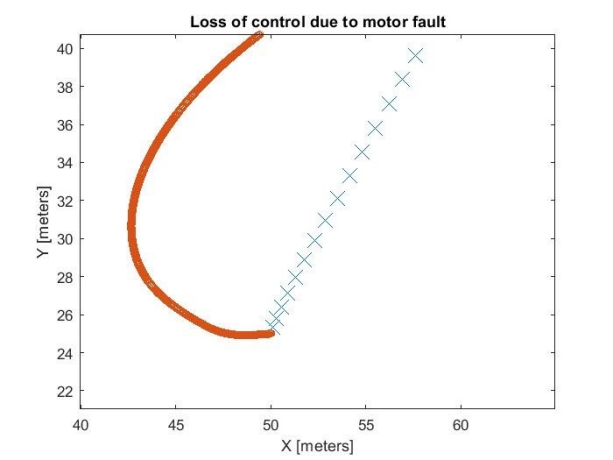
Trajectory 3 : Motor fault with fault-tolerant flight controller

Simulated Octocopter Trajectories

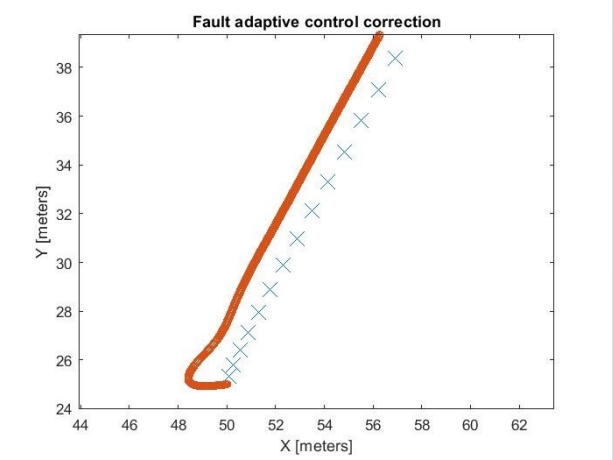
Trajectory 1



Trajectory 2



Trajectory 3



Conclusion

- What went well:
  - Working independently and in person at the lab was helpful
- Challenges:
  - Solving the Synchronicity Problem (Getting Gazebo and our python model to function in sync with each other.
  - Learning socket communication via TCP and UDP
  - Tweaking Digital Twin model to accurately reflect weight and dimensions of real-life drone.
- Lessons Learned:
  - I learn best by doing rather than by observing completed work. (By writing my own code, I conceptualize problems better than when reading other code.)
  - Separating my work space and home space is important mentally.