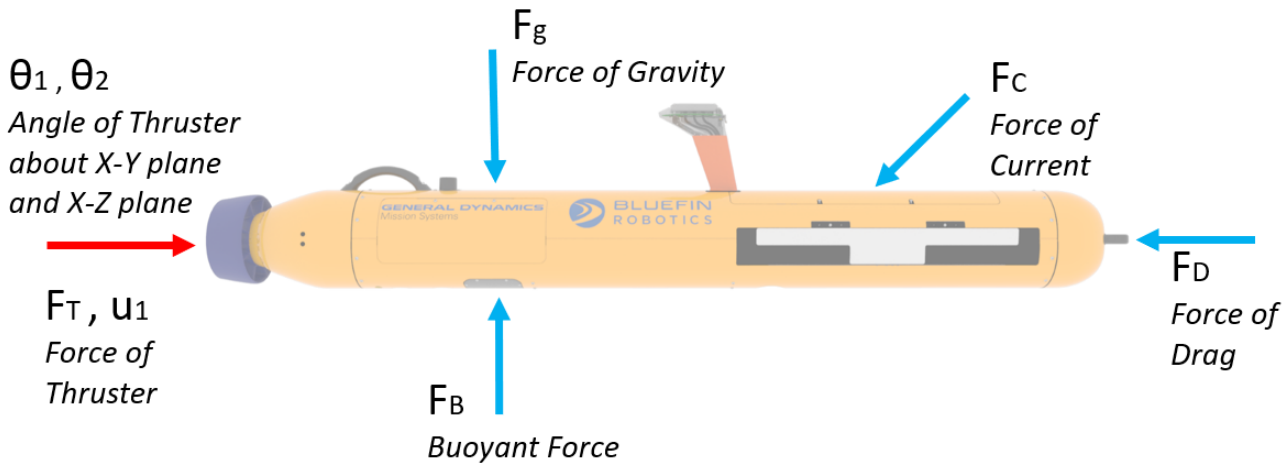
Autonomous Underwater Vehicle Control

RBE 502: Final Project

Sean Tseng, Omri Green, Max Wolfley, Joseph Lombardi

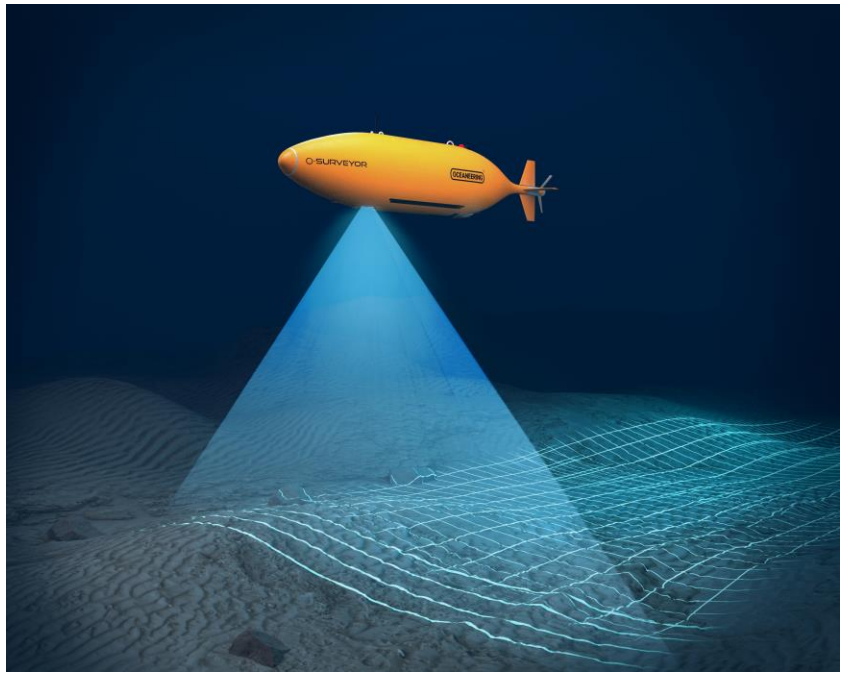


Outline

- Introduction
- Robot Model
- Control Description
- MATLAB Implementation
- Simulation Results
- Discussion
- Conclusion

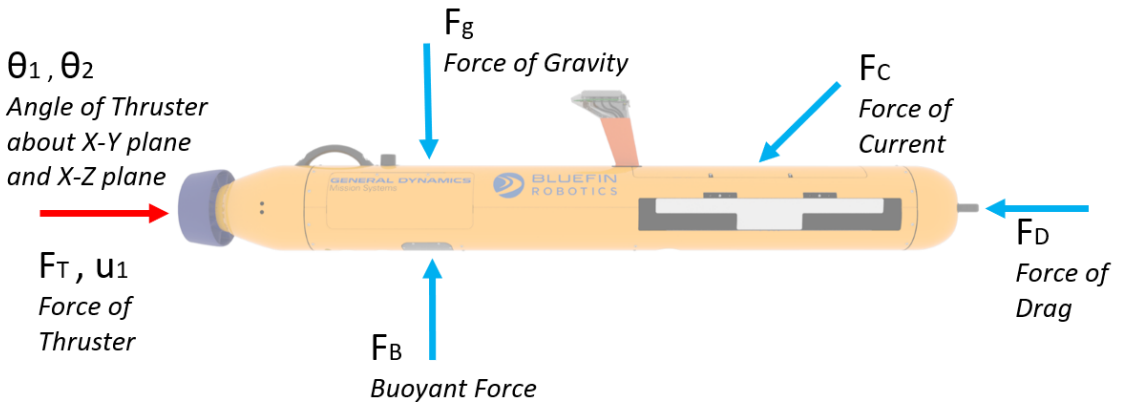
Introduction

- What are AUV’s
- Project Goal



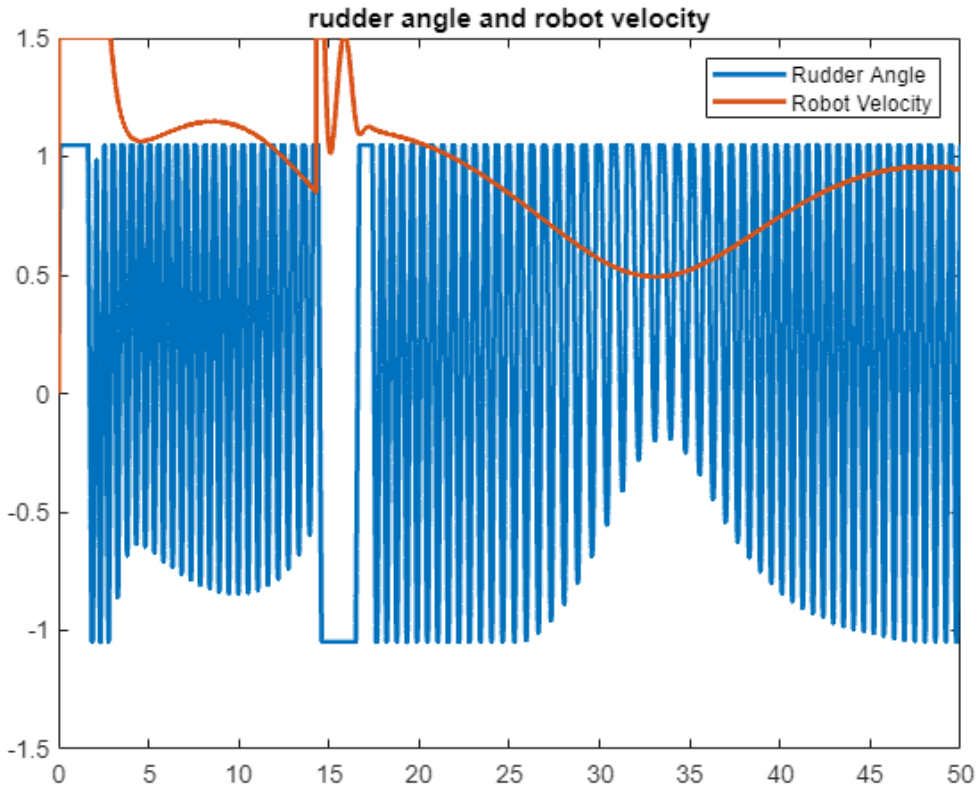
Robot Model

- Model includes:
  - Buoyant force Fb vs. gravity Fg
  - Thruster Ft vs. drag Fd
  - Externalities: Current force Fc



Control Description

- Control:
  - Rudder angle – orientation error
  - Thruster velocity – position error
  - Buoyant force – depth error
- Complete controls for 3D



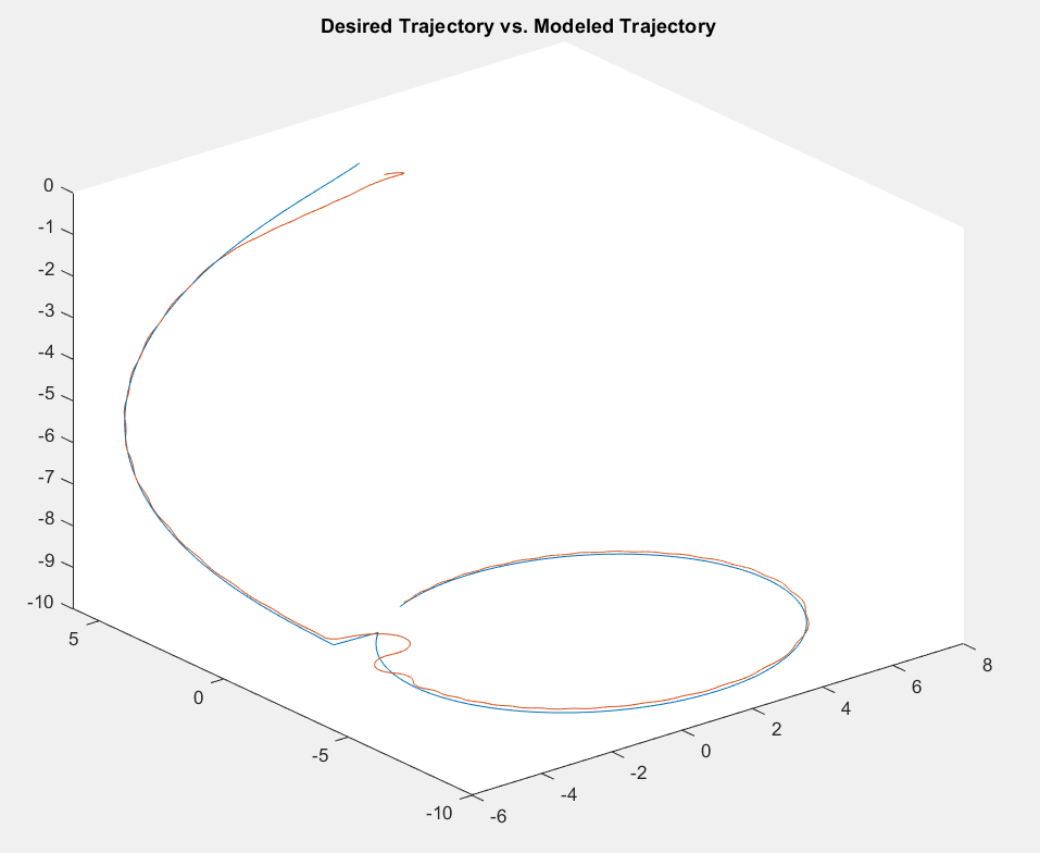
MATLAB Implementation

- Code is discrete time-based (dt = 0.01)
- Robot state iterated over time (50s)
- Desired trajectory over time
  - Initial descent phase
  - Search phase
- Control function (PD)
- Saturation filters
- Iterate to next state

```
%% Initial Configuration of the robot
X(:,1)=[0, 2, 0, 0, 0, 0]; % x, y, z, theta, rudder, velocity
X_error(:,1) = [0,0,0];
X_v_over_t = [];

th_max_fwd = 1.5;
th_max_rev = -0.25;
max_rud_pos = deg2rad(60);
max_z = 1;

th_gain = 4;
z_gain = 4;
r_gain = 400;

fwd_offset = 0.25;
```

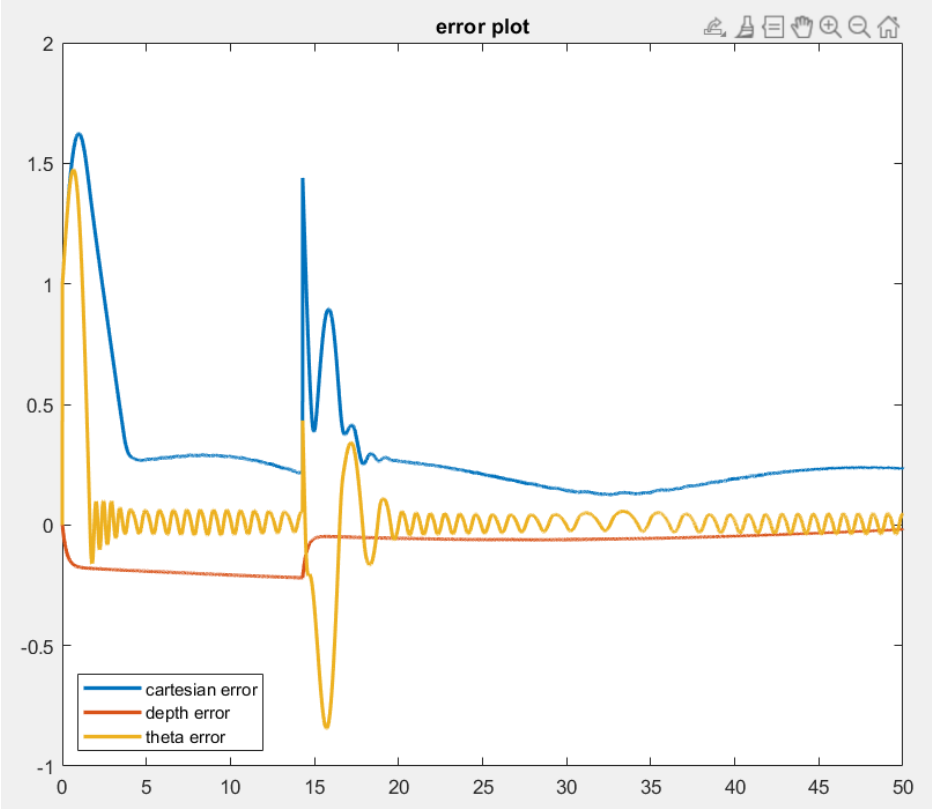
Simulation Results

- Robot starts with position error
- Orients and moves to path
- Descends along spiral
- Follows ‘search’ pattern at depth
- Varying current forces
  - Do not cause large deviations



Discussion

- Cartesian and directional error settle
- Theta error (rudder) oscillates
- Tuning necessary for rudder
  - Currently PD
  - Possibly add integral component



Conclusion

- Successful simulation
- Further development:
  - Tuning control function
  - Advanced model
  - Localization and mapping
  - Arbitrary/disjoint trajectory