

UAV Autonomous Landing

Team Expeditus

Dept. of Computer Science, SDSMT

October 20, 2015

Team Expeditus

Jonathan Dixon, Dylan Geyer, Christopher Smith, Steven Huerta

Sponsor

Dr. Larry Pyeatt

Goal

Software to autonomously take-off, navigate to set waypoints, return to launch pad, and land

Phase Objectives

Phase I

- Build UAV
- Flight Controller Operating Correctly
- Simulation Environment Available

Phase II

- Autonomous landing ready for simulation
- Autonomous landing ready for UAV

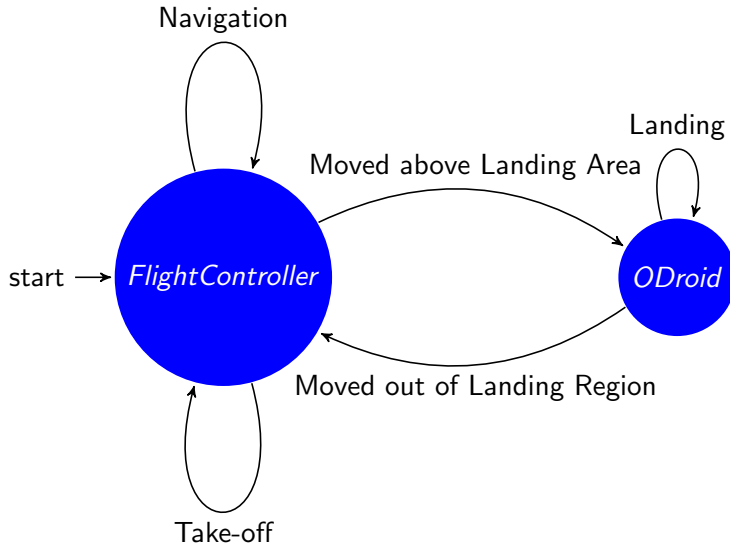
Phase I

- Manual Flight of UAV
- Autonomous Flight of UAV

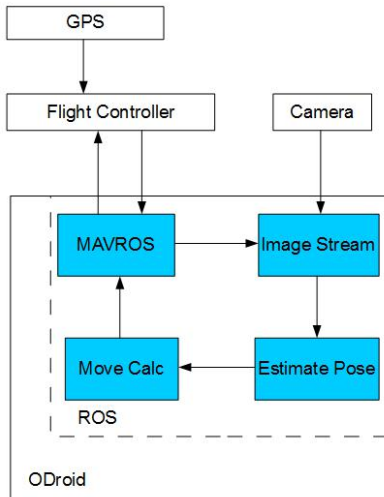
Phase II

- Autonomous Landing in Simulation
- Autonomous Landing of UAV
- Autonomous Take-off, Navigation, and Landing of UAV

Approach - UAV



Approach - Software



Approach - Landing Vision

Put some stuff here about the landing vision approach, maybe a picture or two

Approach - Landing AI

Put some stuff here about the landing ANN approach, maybe a picture or two

Development OS: Ubuntu 14.04

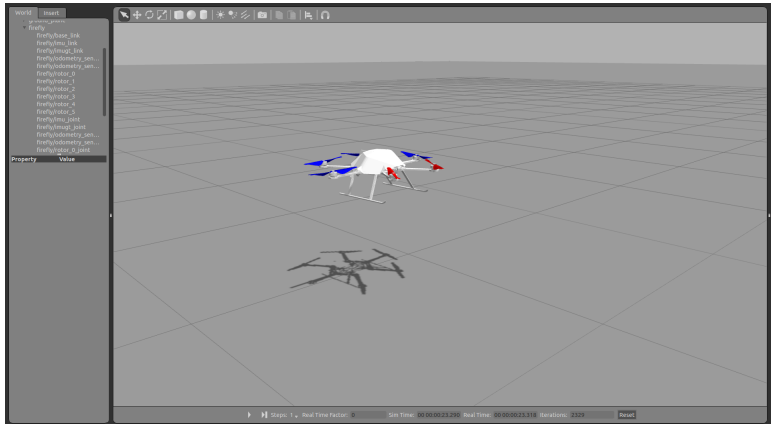
Language: C++

Software Tools

- Robot Operating System(ROS)
- Gazebo
- APM Planner

Simulation & Testing:

- Rotors Sim package - Provides Models for Gazebo



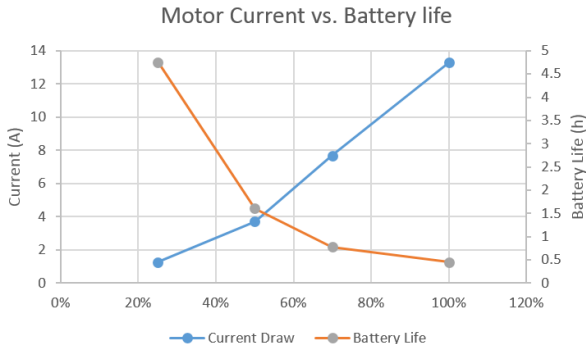
Development - Software Contd.

- MavRos - Communication with Pixhawk through ROS
- Testing - All components will be tested in simulation before being deployed on UAV

```
overmind@overmind:~/landingpad/rotors_sim_ws$
overmind@overmind:~/landingpad/rotors_sim_ws$ rostopic list
clock
diagnostics
/tf
/tf/command/motor_speed
/tf/command/roll_pitch_yawrate_thrust
/tf/gazebo/command/motor_speed
/tf/gazebo/cmd_vel
/tf/gazebo/ground_truth/imu
/tf/gazebo/ground_truth/odometry
/tf/gazebo/ground_truth/pose
/tf/gazebo/ground_truth/pose_with_covariance
/tf/gazebo/ground_truth/position
/tf/gazebo/ground_truth/transform
/tf/imu
/tf/joint_states
/tf/joy
/tf/motor_speed
/tf/motor_speed/0
/tf/motor_speed/1
/tf/motor_speed/2
/tf/motor_speed/3
/tf/motor_speed/4
/tf/motor_speed/5
/tf/odometry_sensor/odometry
/tf/odometry_sensor/pose
/tf/odometry_sensor/pose_with_covariance
/tf/odometry_sensor/position
/tf/odometry_sensor/transform
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/roscpp_log
/rviz
overmind@overmind:~/landingpad/rotors_sim_ws$
```

Hardware Constraints

- 6000mAh Battery
- Power ODroid + Peripherals
- Power 6x DC Motors



Development - Hardware Continued...

Item	Quantity	Total Weight
DC Motor	6	372g
Frame	1	1300g
Battery	1	680g
Camera	2	140g
ODroid	1	48g
GPS Module	1	17g
Total		2557g

1 Motor at 100% produces 970g of lift

Maximum Lift = 5820g

Motors must run at $2557\text{g} / 5820\text{g} = 44\%$

Computational Constraints

- Images: 976 x 582 pixels at ≥ 5 images/sec
- Processing 1 image thus requires $\sim 570,000$ operations
- ODroid has 8 cores at 1.4 GHz
 - Ideal throughput ~ 10 Billion operations/sec

Cost

Build 1		Build 2	
Item	Cost	Item	Cost
Controller	\$199.99	Controller	\$199.99
ODroid	\$75.95	ODroid	\$75.95
Sensors	\$167.23	Sensors	\$167.23
Frame Kit	\$242.48		
Power Kit	\$119.98		
Radio Set	\$100.00		
Extra Parts	\$95.15		
TOTAL	\$1000.78	TOTAL	\$443.17

General

- Review previous iteration documentation & code
- Begin pilot training for manual control
- Review Landing Pad model with Landing Pad teams

Setup Development Environment

- Ubuntu 14.04
- Gazebo/Rviz
- ROS - Jade Distro

Inspect Current Quadrotor

- Identify missing or non-functioning components
- Generate order list

Risk

- Reliance on Flight Controller
- Dependency on external team for Landing Pad
- No UAV Backup

Setbacks

- Non-functional components
- Little carry-over from previous year

Conclusion

Conclusion-y stuff here

Questions?