CRATES: Cognitive Robotics Architecture for Tightly-coupled Experiments and Simulation

Andrew Symington
University College London
a.symington@ucl.ac.uk

ROSCON 2014

Introduction

- My position in CompLACS
- Coordinating multiple agents in order to achieve some complex task (through decomposition)
- EU-based collaboration
  - Theorists
  - Engineers
  - ERC

- Quadcopters
- Theory → Simulation → Experiments

Simulation: QRSim

- Experiments = time + risk
- QRSIM (to appear in IROS'14)
- Features
  - Platform dynamics
  - Environment noise
  - Sensor noise
- Issues
- CRATES: migration to ROS

Simulating Quadrotor UAVs in Outdoor Scenarios

Andrew Symington, Renzo De Nardi, Simon Julier and Stephen Hailes
{a.symington,r.denardi,s.julier,s.hailes}@ucl.ac.uk

***Abstract*—Motivated by the risks and costs associated with outdoor experiments, this paper presents a new multi-platform quadrotor simulator. The simulator implements a novel second-order dynamic model for a quadrotor, produced through evolutionary programming, and explained by domain knowledge. The model captures the effects of mechanics, aerodynamics, wind and rotational stabilization control on the flight platform. In addition, the simulator implements military-grade models for wind and turbulence, as well as noise models for satellite navigation, barometric altitude and orientation. The usefulness of the simulator is shown qualitatively by a comparing how coloured and white position noise affect the performance of offline, range-only SLAM. The simulator is intended to be used for planning experiments, or for stress-testing application performance over a wide range of operating conditions.**

***Index Terms*—Unmanned Aerial Systems, Simulation**

I. Introduction

Unmanned aerial vehicles (UAVs) have received significant research attention as a result of the numerous military and civilian applications that they enable. Quadrotors have proven themselves as favourable platforms for research, since they are inexpensive, easy to fly, agile and safe to operate [1].

Conducting outdoor experiments with quadrotors requires ideal weather conditions and compliance with local aviation regulations, uses significant resources, and puts platforms at risk. It is therefore preferable to evaluate scenarios in simulation prior to experimentation, where possible. However, the suitability of such an approach depends largely on the fidelity of the simulator. The work in this paper is therefore motivated by the need for a quadrotor simulator that (i) enables multiple platforms to be simulated at the application level, (ii) provides realistic models for dynamics, wind and sensor noise, and (iii) balances accuracy with speed, thereby enabling real-time, or near real-time, performance.

A great number of simulators exist as tools to for training radio control enthusiasts, but not as platforms for research. Research simulators based on first principles have been developed for USARSim [2] and Gazebo [3]. Both capture the behaviour of an ideal platform's mechanics and aerodynamics, but not of the low-level controller used to perform rotational stabilization control. The design of the stabilization control algorithm is usually proprietary, and therefore cannot be modelled by first principles. Although effort has been made to investigate the effect of wind and turbulence on quadrotor dynamics [4], such research remains to be integrated into the widely-used simulators. More recent simulators are based on the Robotic Operating System (ROS) and include SwarmSim X [5] and Hector Quadrotor [6], which use a PhysX and

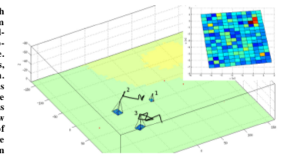


Figure 1. A screen shot of a QRSim search and rescue simulation.

Gazebo as simulation cores respectively. The contribution of this work is the open-source[1] QRSim simulator depicted in Fig. 1 that distinguishes itself by:

1) A general second-order dynamic model that captures the behaviour of mechanics, aerodynamics, wind and stabilization control. The model is derived from flight data using the co-evolutionary programming method, with model parameters fitted to a specific platform.
2) Models for environment and sensor noise, including a military-grade model for turbulence that is validated against a third-party implementation, and a satellite navigation model that is validated against raw data.
3) Implemented as self-contained Matlab code that is controllable over a TCP/IP interface, with the goal of being a modular and extensible tool for research.

The remainder of this paper is structured as follows. Sec. II discusses the dynamic model used in the simulator, and shows how model parameters were obtained for a specific flight platform. Sec. III then describes the environmental and sensor noise models, as well as how they were validated. Sec. IV then provides a qualitative illustration of the usefulness of the simulator by comparing the effect of white and coloured sensor noise on offline, range-only simultaneous localization and mapping (SLAM) for a single platform. Sec. V concludes the paper, and discusses directions for future work.

II. Quadrotor Dynamics and Control

A flight control system (FCS) performs sensing to determine orientation or position, and then selects corrective motor changes to stabilize the platform [1]. Orientation is usually

[1]Source code is available at https://github.com/UCL-CompLACS/qrsim

Simulation: CRATES

hal_quadrotor

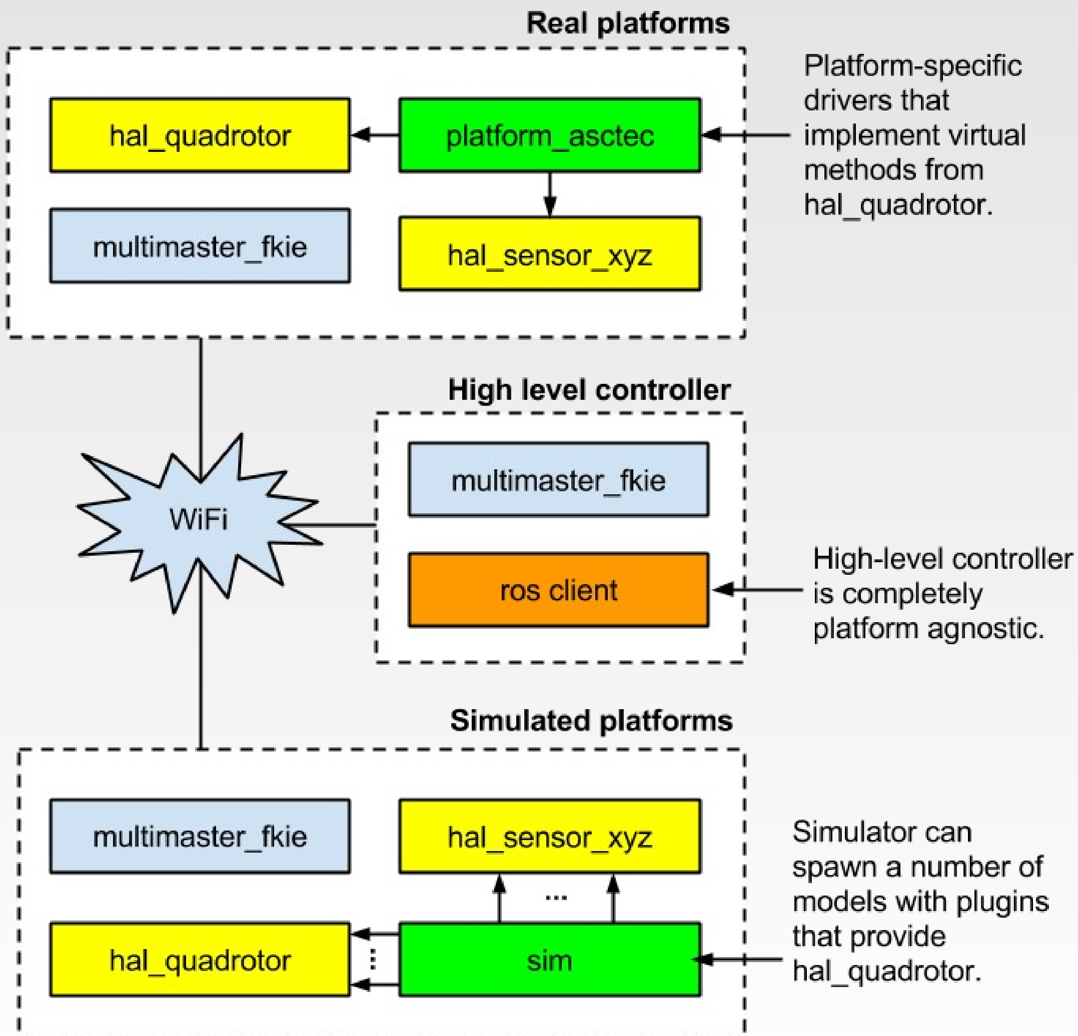
Topics
- Estimate
- Control
- Truth

Services
- Waypoint
- Velocity
- VelocityHeight
- AnglesHeight

hal_sensor_xyz

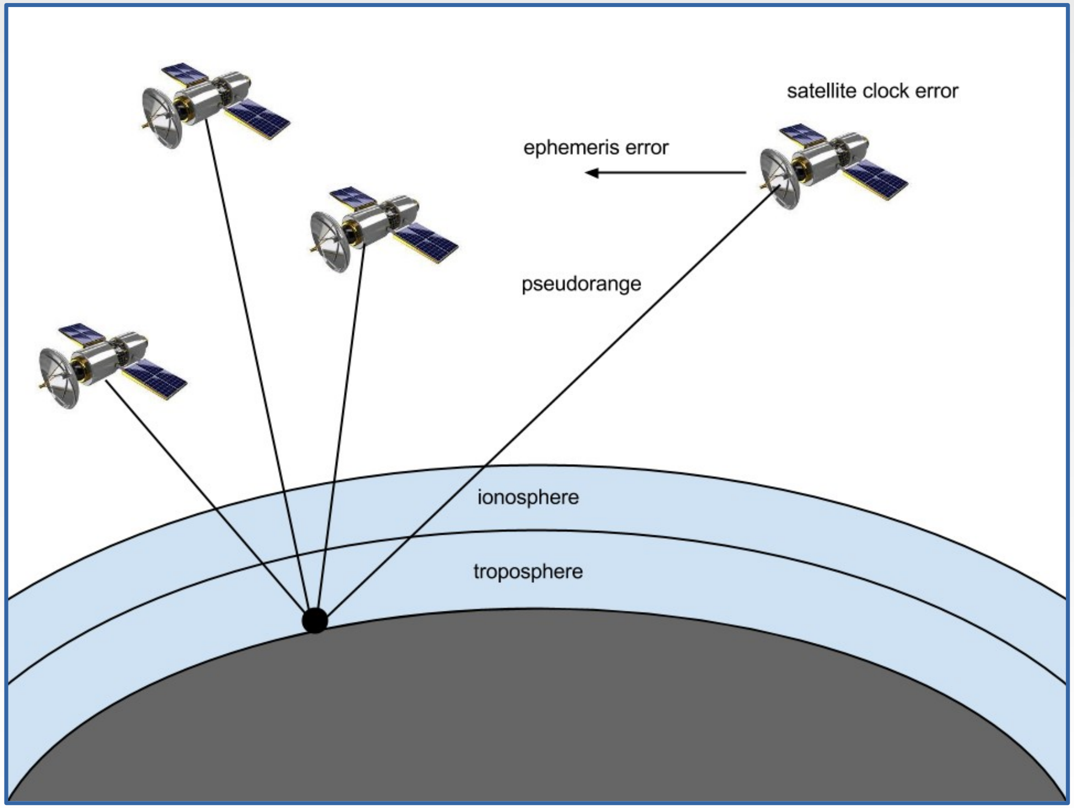
Topics
- Data

Services
- SetRate

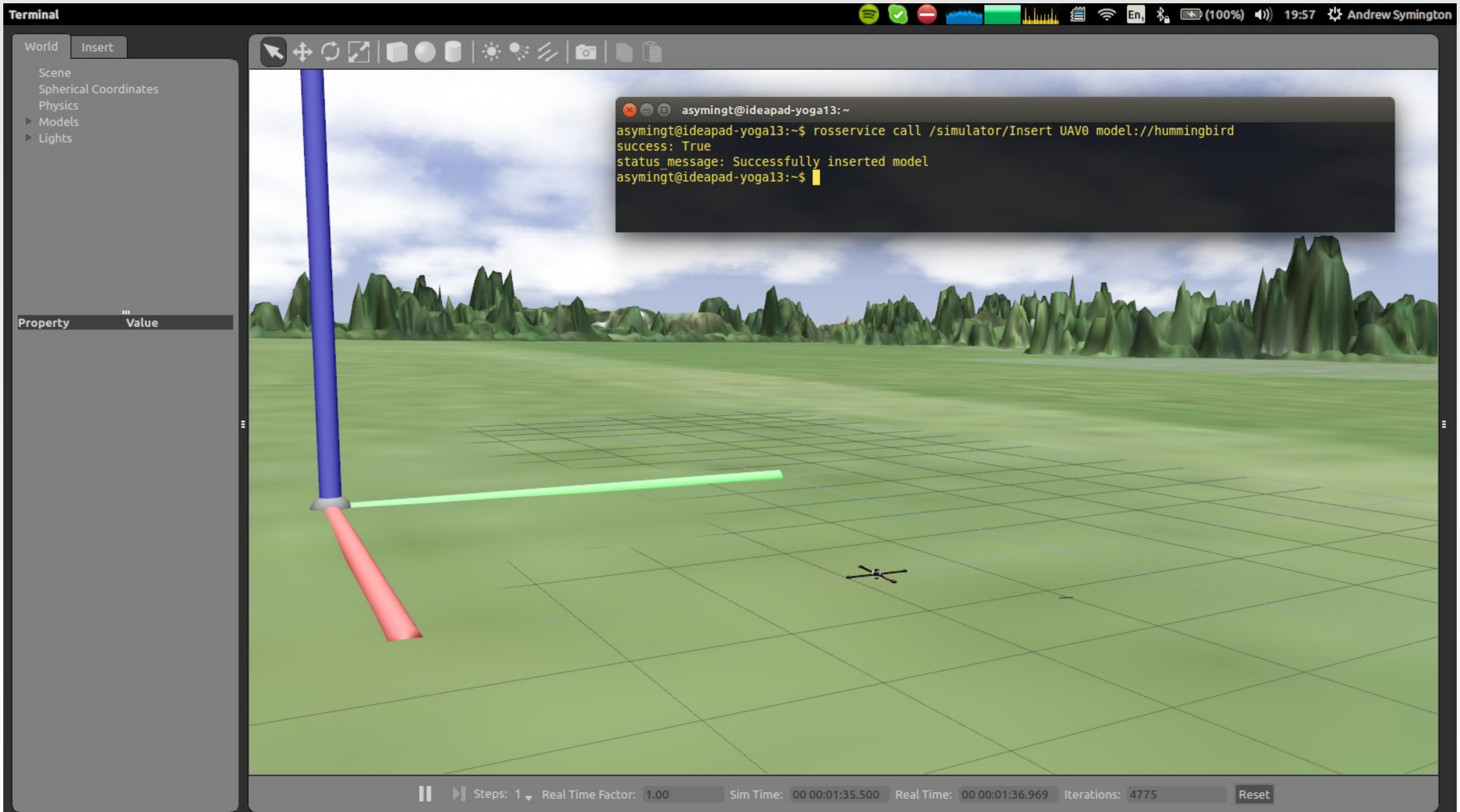


Simulation: Improved GNSS model

- GNSS performance → controller performance
- Time-correlated
  - Ephemerides
  - Satellite clock
  - Receiver clock
  - Ionosphere
  - Troposphere
- New GPS module
- IGS traces



Simulation: CRATES

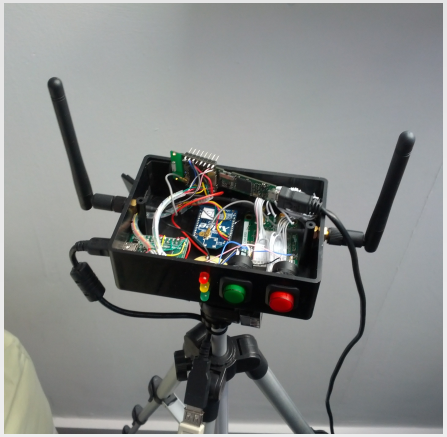


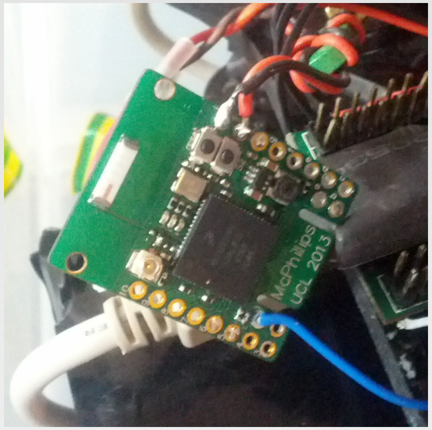
Hardware: platform overview

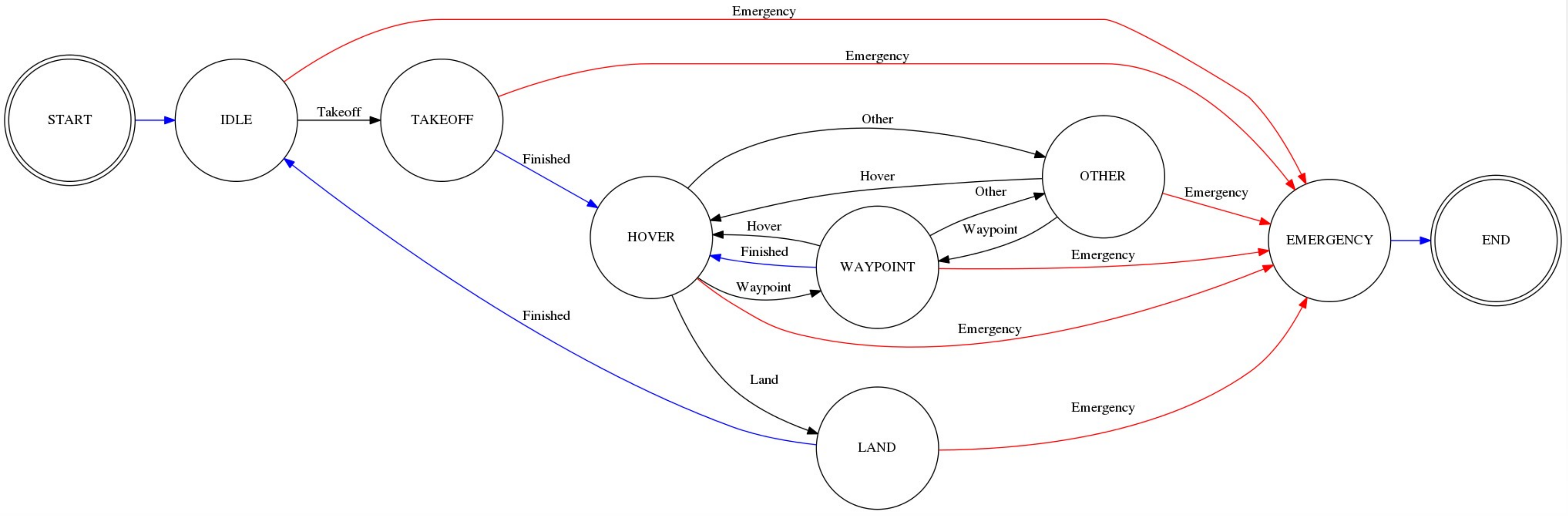
- Ascending Technologies Pelican (5)
- Platform limits
  - Battery < 20minutes
  - Max 5m/s velocity, 200deg/s yaw rate
  - 650g max payload
  - 10Hz < measurements / control < 10Hz
  - Firmware flashing not permitted
- Supplied with manual transmitter
- Added a 2.0Ghz Atom “SBC”

Hardware: safety

- Levels
  - Manufacturer
  - Finite state model*
  - Safety module*



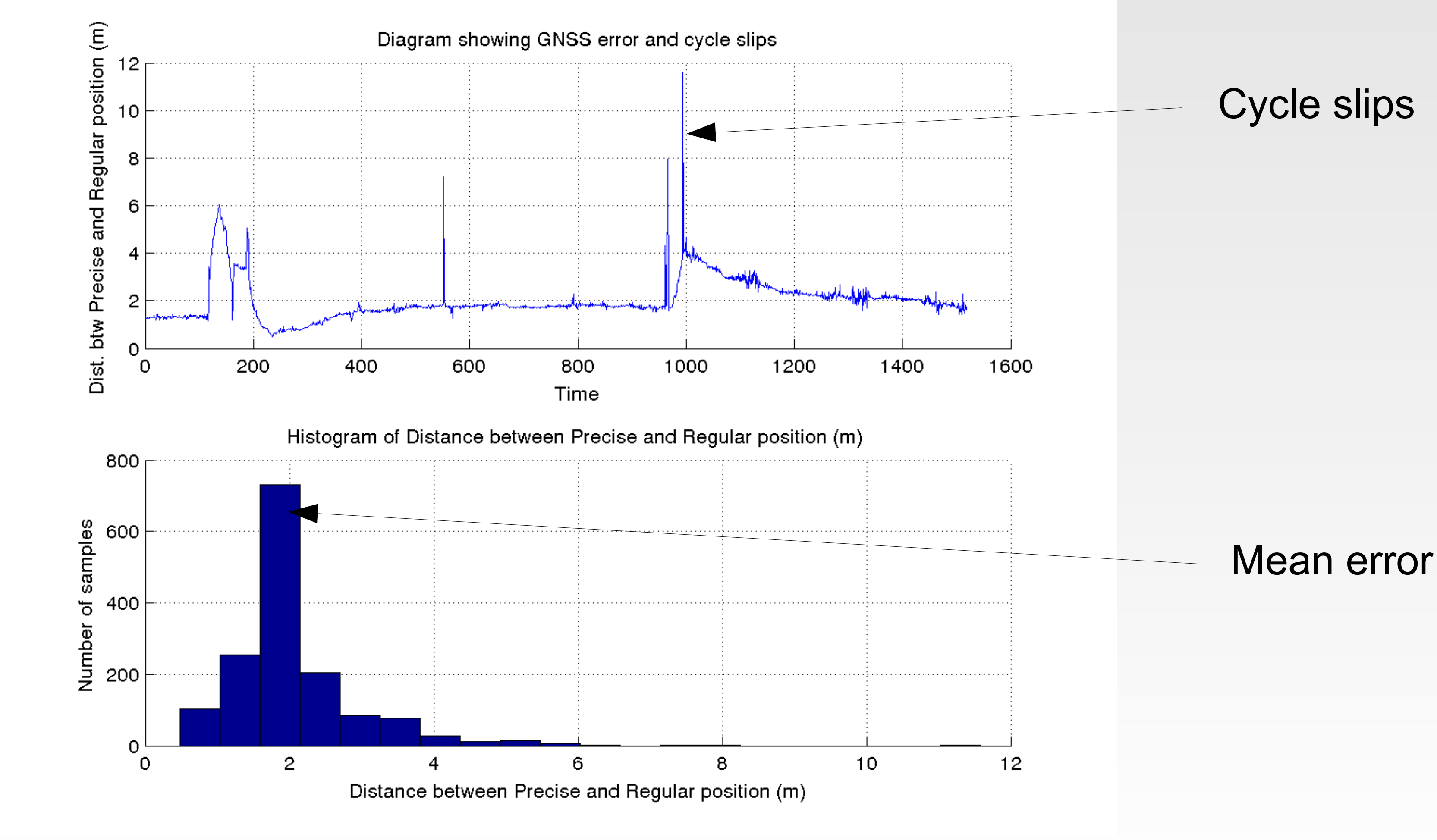




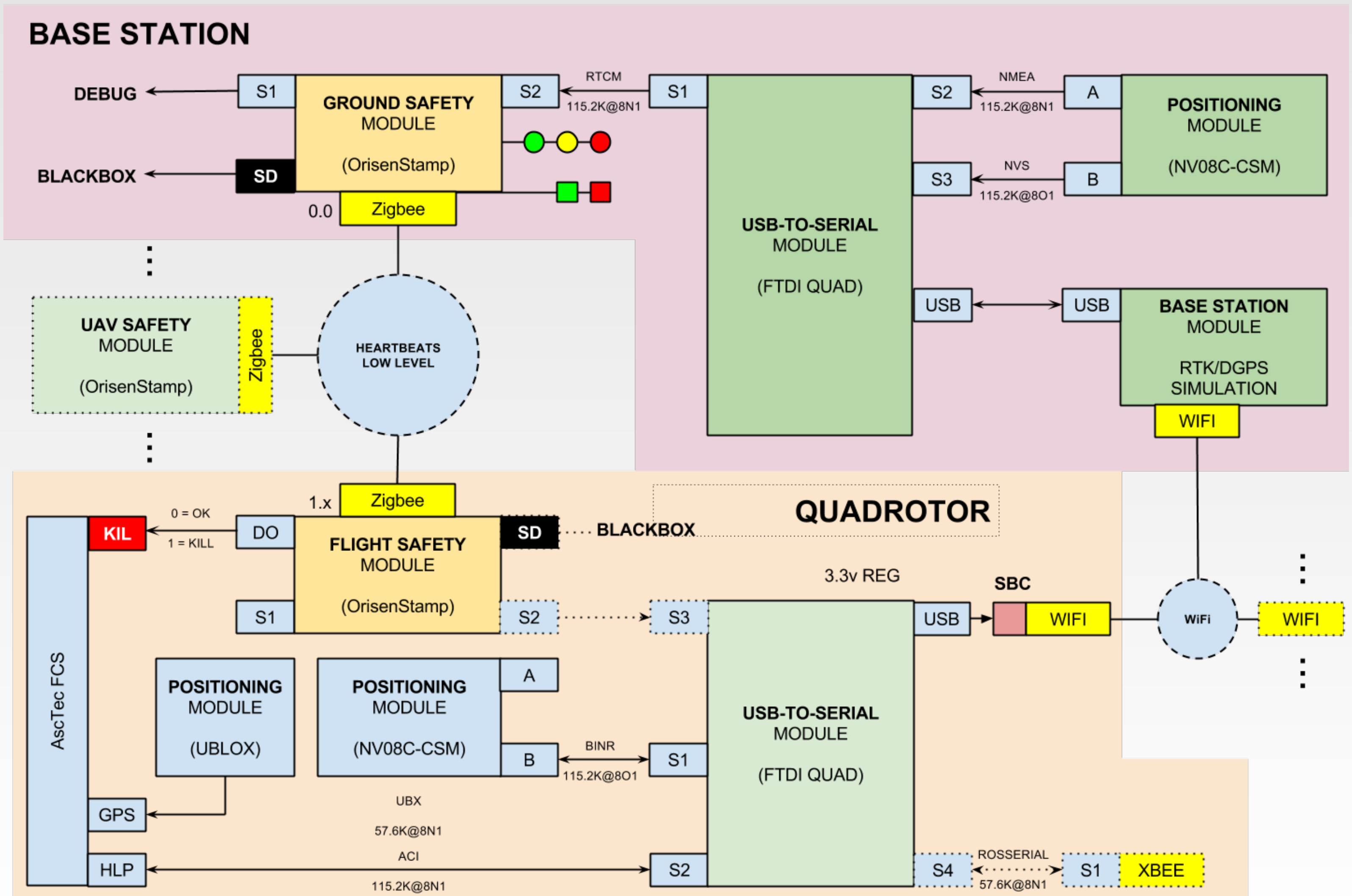
Hardware: ground truthing

- Goal: accurate position
- Errors
  - Code-phase: ~meters
  - Carrier-phase: ~decimeters
- Cycle slip
- Hardware
  - NV08C-CSM-BRD
  - Tallysman GPS/GLO

Hardware: ground truthing



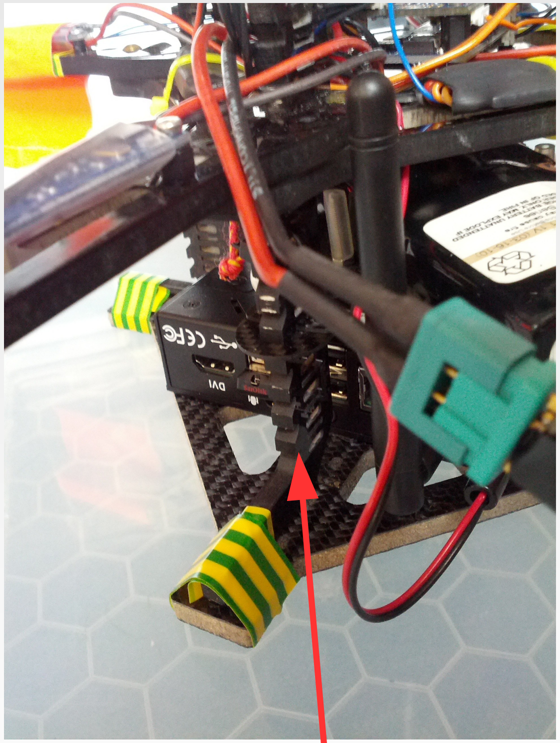
Hardware: putting it all together



Hardware: mechanical issues

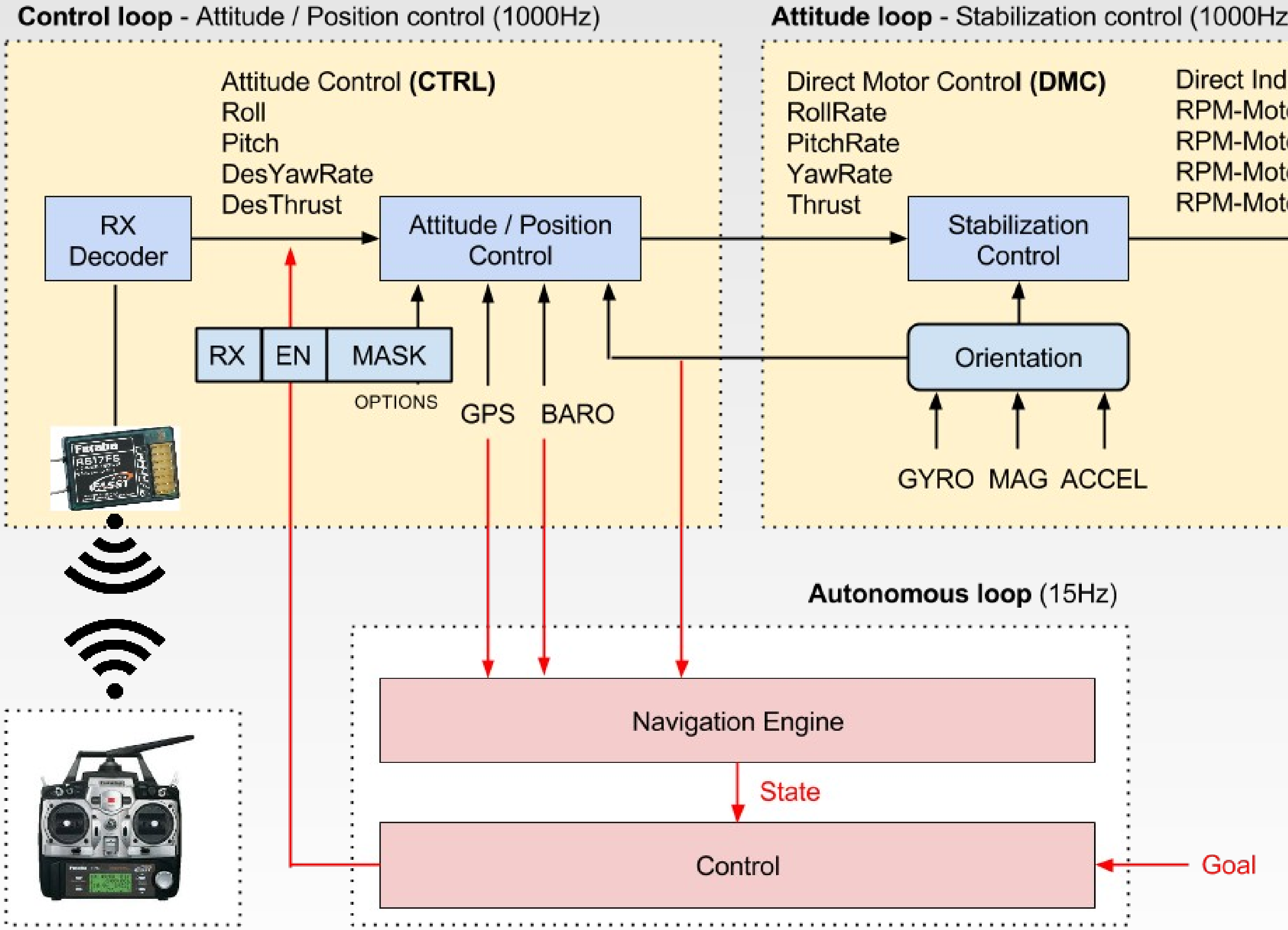
Propeller fatigue

Leg fatigue



CNC legs + fittings

Hardware: black box FCS



Possible goals
- Waypoint, Yaw
- 3D Velocity, Yaw
- 2D Velocity, Height, Yaw
- Angles, Height
- Takeoff (to altitude)
- Land
- Hover (stay at current point)

Theory: Path Integral Control

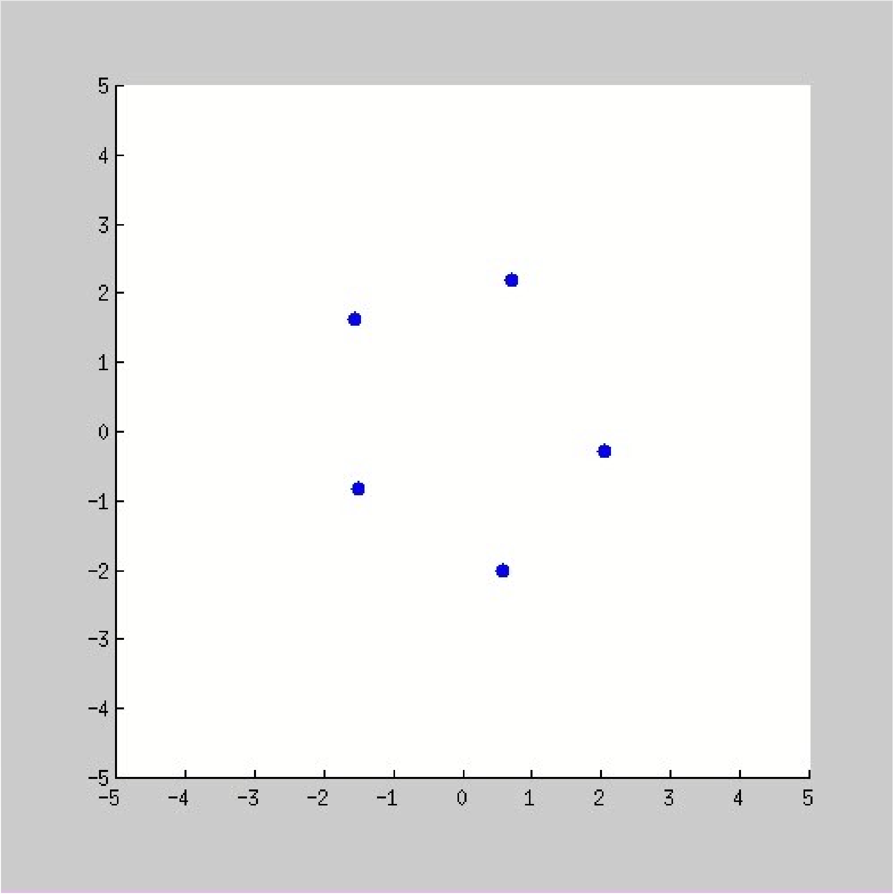
- Coordinate actions of multiple quadrotors
- Path Integral Control
  - **COST:** Complex task is specified through a cost function
  - **DYNAMICS:** Controlled noisy movement
  - **CONTROL:** Given the current state the optimal control is computed with sampling methods, based on rollouts
- Approach
  - **FEEDBACK**: Control is re-computed after state update
  - **SAMPLING**: Computations based on a simplified model

Theory: Holding Pattern

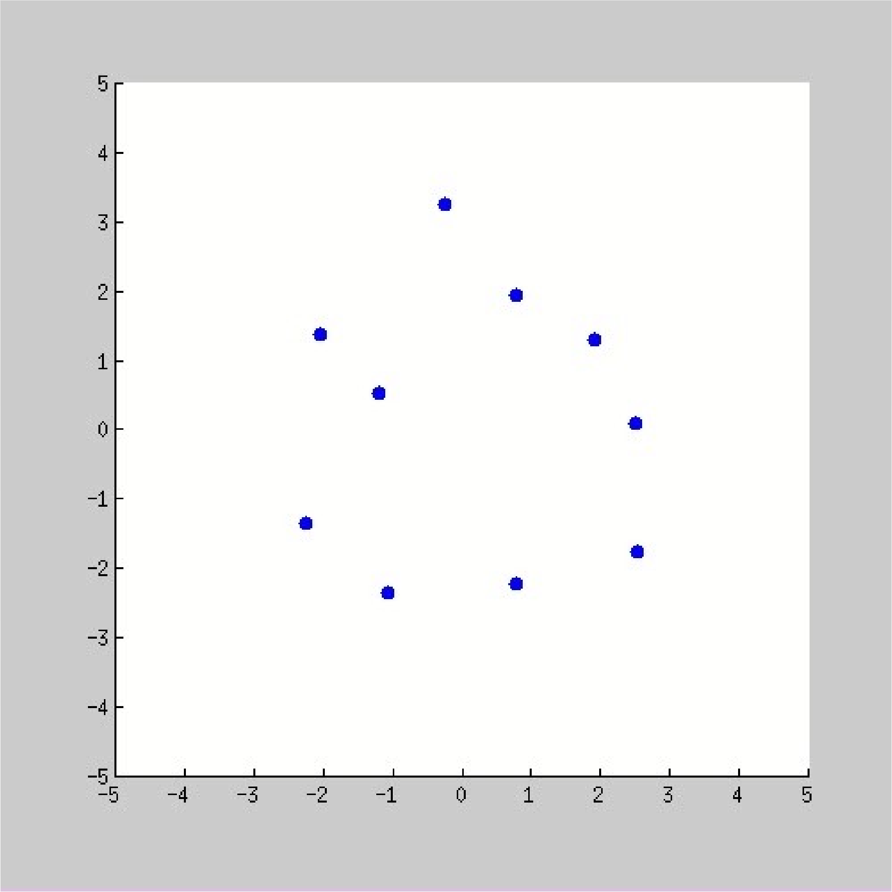
- Task
  - Aircraft queue up for landing above a landing zone
  - They must maintain a safe distance from each other
  - Aircraft cannot hover; must maintain a minimum velocity

- Cost
  - Cost for too low or too high velocity
  - Cost for pair-wise distance
  - Cost for distance to the landing zone
  - Cost for collisions

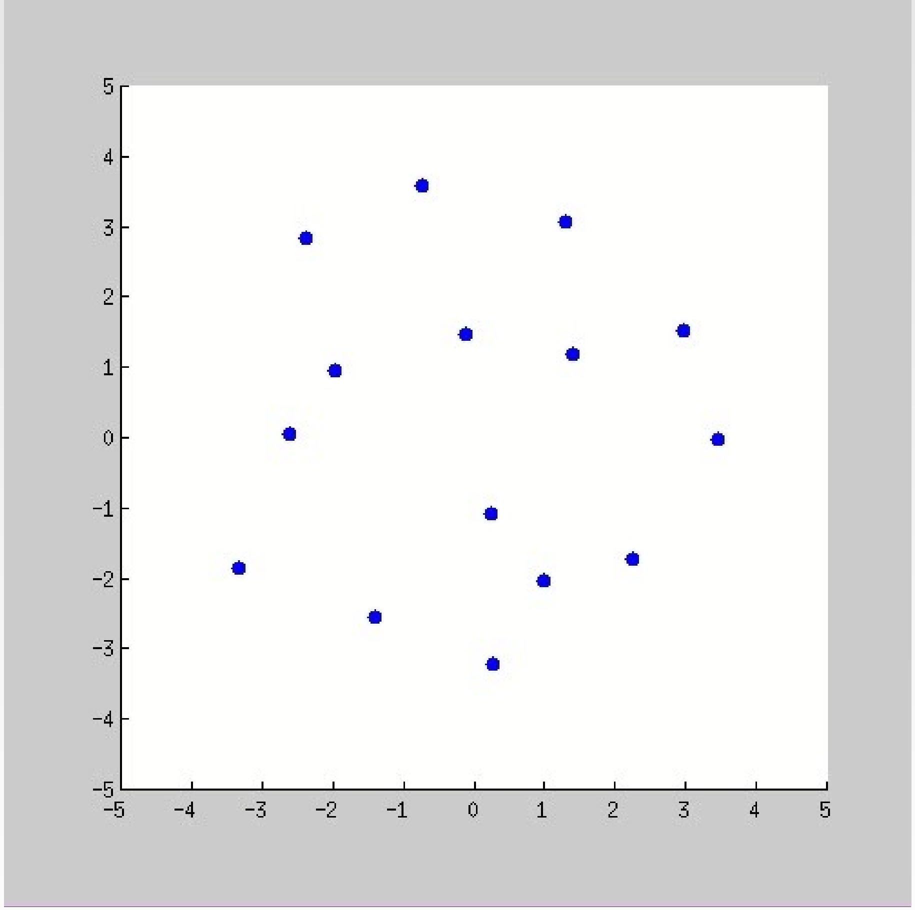
Holding Pattern : 5 platforms



Holding Pattern : 10 platforms



Holding Pattern : 15 platforms



Results

- Demo

Experimental results

Conclusion

- Integration with mav_tools and hector_quadrotor
- HAL-based abstraction
  - Does this generalise?
- Some questions about ROS
  - Command-line tool latencies
  - Message inheritance
  - Multimaster integration plans
  - Quality of Service guarantees

All code available open source: https://bitbucket.org/asymingt/crates