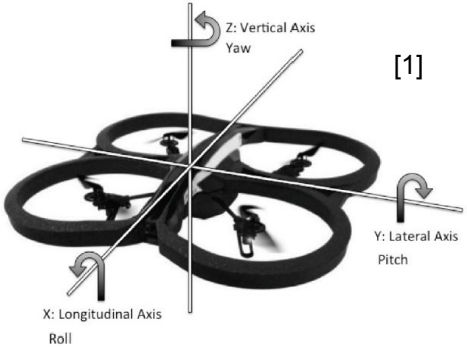
OCTODRONE

Simulation and Deployment for Autonomous Drone Networks

Overview

- Background on quadcopters
- Project introduction
- The simulation software
- octoDrone demonstration
- Research implications
- Deployment demonstration
- Outreach implications
- Wider uses & further work
- Questions

Background on Quadcopters



[1]

- General name for any vehicle that achieves flight using four rotors
- More usually refers to unmanned drones that follow the above configuration
- Risen to prevalence within the last 5 years
- Mostly manually controlled.
- Movement is achieved through manipulating the rate of rotation of the four blades, allowing for movement in any direction as well as turning
- Autonomous applications usually limited to pre-programmed movements or arrangements of several pre-programmed movements.
- Little way of making drones perform arbitrary tasks.

Project Introduction

- Initially, building a system to solve problems with drones.
- We discovered the tools available were inaccessible and bloated.
- Decided to instead create a simulator, specifically for drone networks.

- Extension: Added visualisation (similar to NetAnim).
- Extension: Physical deployment to real drones.

octoDrone

- Discrete event network simulator for unmanned aerial vehicles.

- Accessible to academics and students.

- Extendable to wide array of applications.

The Team

Will Seymour - Project Manager, communications, and physical deployment.

Alex Henson - Coordinating the report and research.

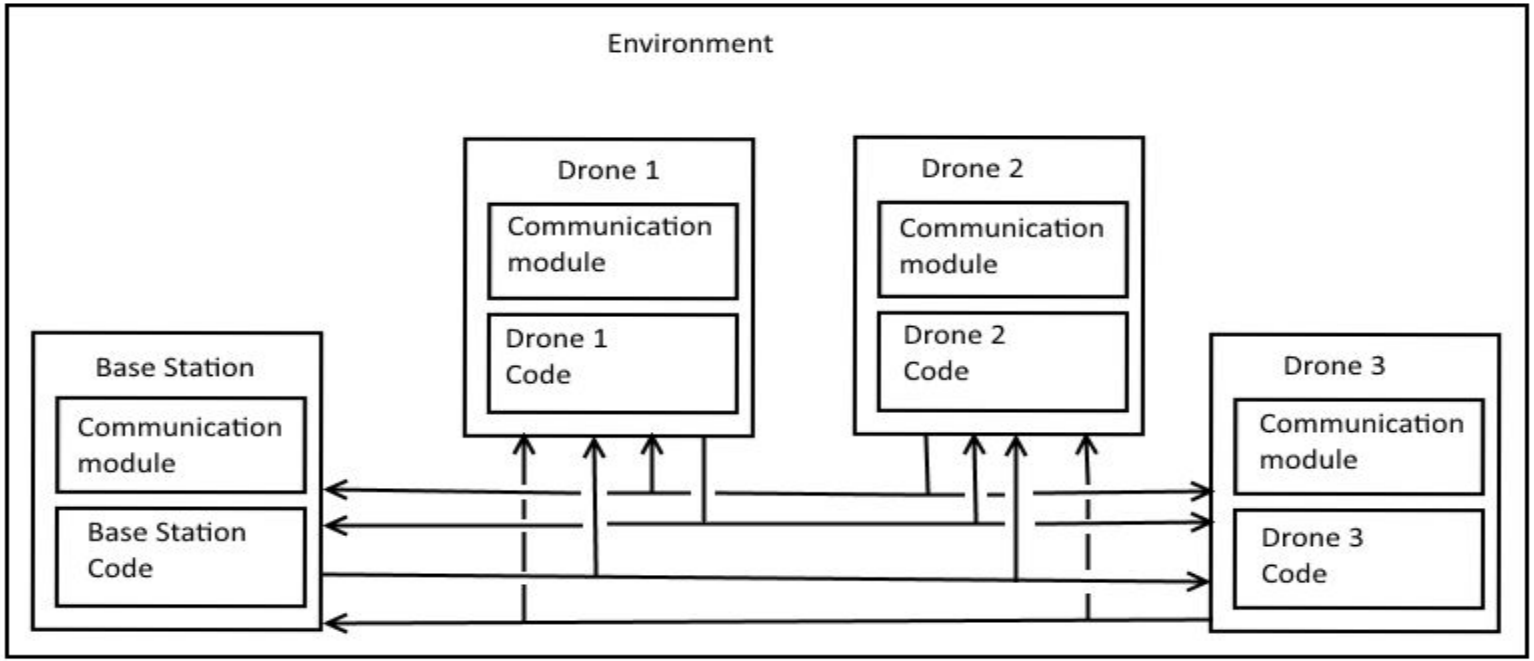
Ben De Ivey - Developing the demonstration simulation.

Jon Gibson - Developing the simulator.

Simulation Software: Overview

- Written in C++
- Focus on emulating multiple systems through the use of threads.
- Minimal feature set so arbitrary drone programs can be run on the simulator
- Primary focus was flexibility and extensibility.

Simulation Software: Structure Example



Simulator Software: Structure

- Environment acts as the container for the rest of the simulator
- Each component of the simulation exists within the environment
- Each component consists of two parts: a program and a communication module.
- The communication module handles the “how” of networking
- The program handles the “what” of the component’s actions
- Each component is capable of broadcasting to all others in range, it is decided by each other component’s communication module to accept or reject messages based upon whether they are relevant to the attached component.
- Communications modules and programs are modular

Simulation Software: Visualisation

GLFW

- Works via a dedicated draw thread
- Separate notions of a draw frame and a simulator tick
- Components of the simulator push elements to be drawn for a number of ticks (typically 1 for drones,as they are pushed every tick, and more for broadcasts as messages need to linger so they are visible)
- At the end of each simulator tick (when all drone movement for that timestep has finished) the visualiser steps
- After each step, any elements that no longer have to be drawn are removed from the list of elements.

octoDrone Demonstration

Creating a Program: API

Send a message:
```
void send_message(Message* contents);
```

Blocking Receive:
```
Message* wait_for_message();
```

Non-blocking receive:
```
void receive_message(std::string contents);
```

Get position:
```
double getX();
double getY();
double getZ();
double getSpeed();
double getAngle();
```

Movement:
```
void turn(double angle);
void move(Direction direction, double speed, double distance);
```

Feedback:
```
bool isAlive();
bool has_finished_moving();
double sense(std::string type);
```

User implemented:
```
virtual bool message_callback(Message* message) = 0;
virtual void run() = 0;
```

Creating a Simulation

```
std::map<std::string, data_type>* sensor_map = new std::map<std::string, data_type>;
Environment* env = new Environment(*sensor_map, 1.0, false);
std::atomic_flag stdout_lock = ATOMIC_FLAG_INIT;
IpAllocator allocator = IpAllocator(10, 0, 0, 1);


CommMod* comm_basic1 = new Basic_addressed(env, &stdout_lock, allocator.next());
CommMod* comm_basic2 = new Basic_addressed(env, &stdout_lock, allocator.next());
CommMod* comm_basic3 = new Basic_addressed(env, &stdout_lock, allocator.next());
CommMod* comm_basic4 = new Basic_addressed(env, &stdout_lock, allocator.next());
CommMod* comm_basic_base = new Basic_addressed(env, &stdout_lock, "10.0.0.255");

SensingBaseStation* basestation = new SensingBaseStation(comm_basic_base, 0.0, 0.0, 0.0, 1.0, 1.0, 10.0, 10.0);
env->setBaseStation(basestation);

SensingDrone* drone1 = new SensingDrone(comm_basic1, 1.0, 1.0, 0.0, 0.1, 0.5, env, false);
SensingDrone* drone2 = new SensingDrone(comm_basic2, 2.0, 2.0, 0.0, 1.0, 0.5, env, true);
SensingDrone* drone3 = new SensingDrone(comm_basic3, 2.0, 3.0, 0.0, 1.0, 0.5, env, true);
SensingDrone* drone4 = new SensingDrone(comm_basic4, 2.0, 4.0, 0.0, 1.0, 0.5, env, true);

env->addDrone(drone1);
env->addDrone(drone2);
env->addDrone(drone3);
env->addDrone(drone4);

env->run();
```

Command Line Output

```
basic_addressed@10.0.0.1:  waiting for new area
basic_addressed@10.0.0.4:  waiting for new area
basic_addressed@10.0.0.3:  waiting for new area
basic_addressed@10.0.0.2:  waiting for new area
Num Drones: 4
basic_addressed@10.0.0.1: NEWAREA=1.000000,1.000000,3.250000,10.000000
basic_addressed@10.0.0.1: Exiting
basic_addressed@10.0.0.4: NEWAREA=3.250000,1.000000,5.500000,10.000000
basic_addressed@10.0.0.4: Exiting
basic_addressed@10.0.0.255:  Received point (1.000000, 1.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 2.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 3.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 4.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 5.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 6.000000, 1.000000, 42.000000 from drone 10.0.0.1
basic_addressed@10.0.0.255:  Received point (1.000000, 7.000000, 1.000000, 42.000000 from drone 10.0.0.1
```

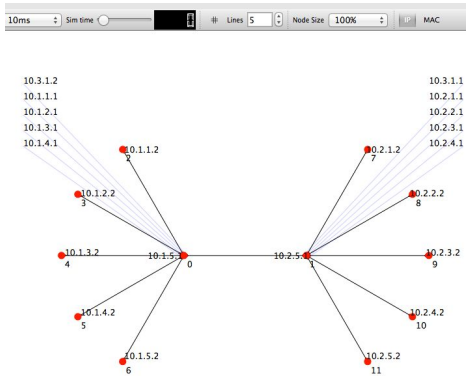
Research Implications

Examining each aspect of the network simulator, the implications for its use in research are as follows:

- Enabling research into sensor networks
- Developing communications routing algorithms
- Developing libraries for controlling quadcopter hardware
- Real-world testing based on research

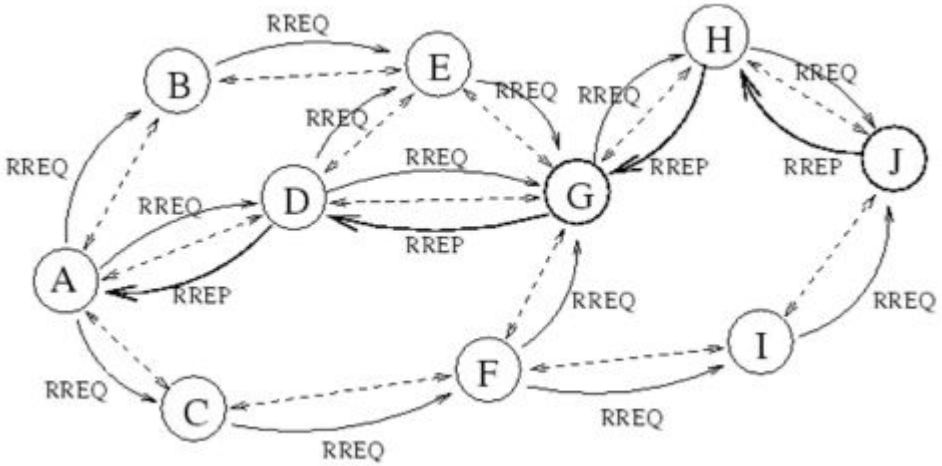
Research into Sensor Networks

- Other network simulators such as ns-3 allow for research but are relatively static
- But octoDrone specifically remains open to development
- Simplified enough to focus purely on required functionality for extension



Developing Communications Routing Algorithms

- Simulator comes with default library currently, but can be expanded easily (modularity)
- Libraries are low-level enough to precisely specify requirements
- Lightweight enough to make even complex simulations fast

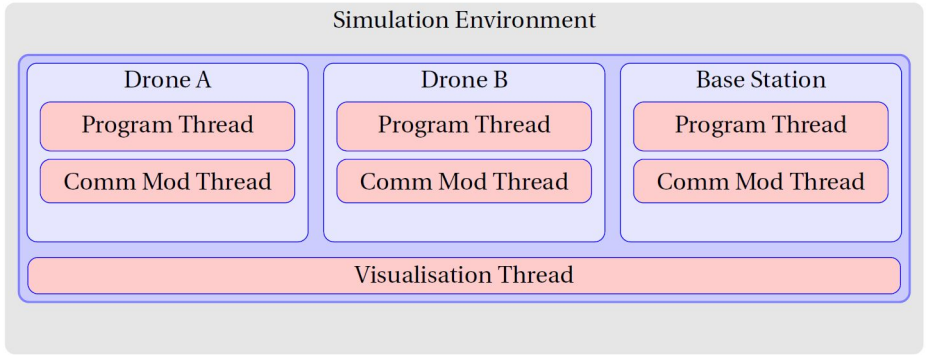


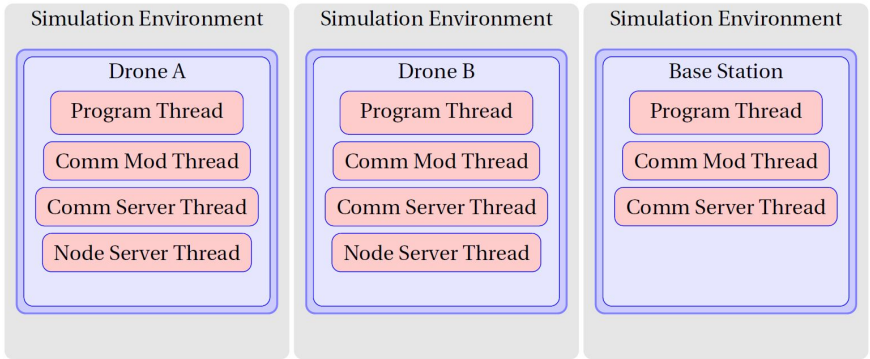
Real-world Testing Based on Research

- The simulator is also a framework for deployment on drones
- octoDrone allows for the creation of libraries for different hardware
- This allows for easy testing of protocols and algorithms in the real world
- Allows researchers to ascertain how well a protocol functions outside of ideal conditions

Deployment Demonstration

Architecture Comparison






TP-LINK
TL-SG105


DELL


DELL

Deployment: Console Output

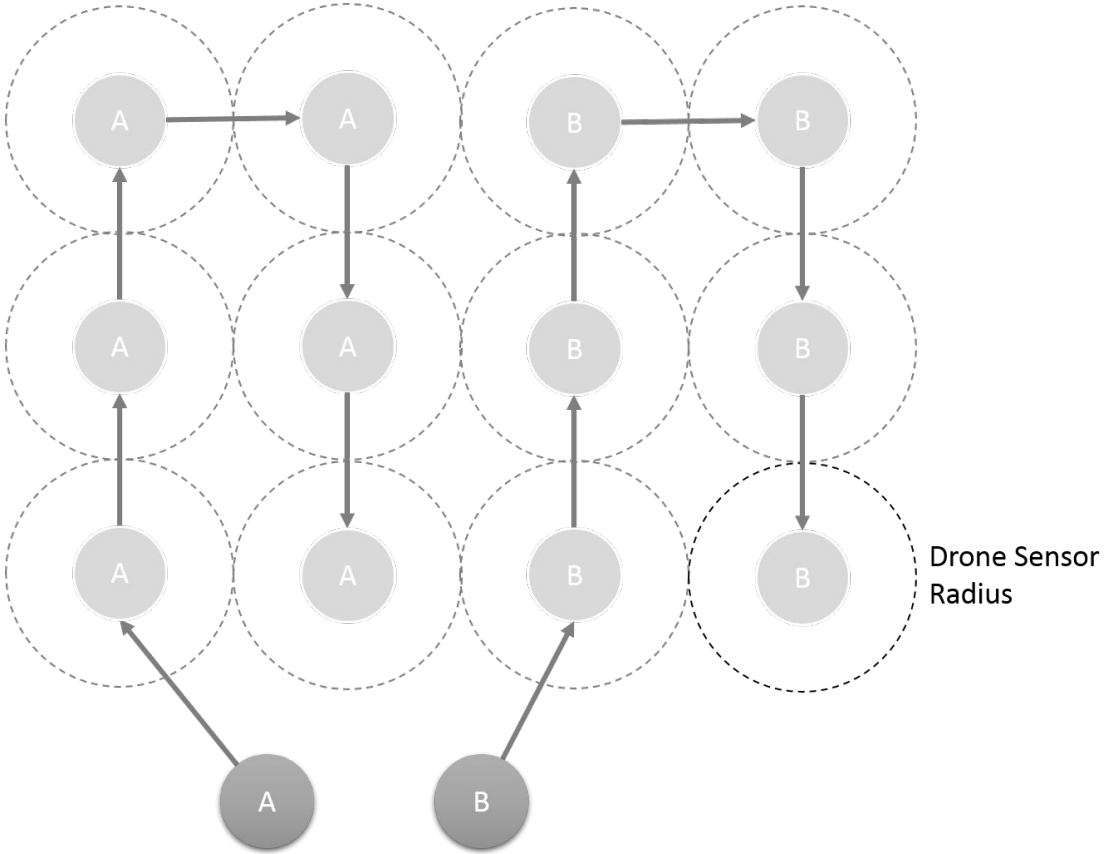
```
init@nodeServer: listening on socket ./parrot.sock
send@nodeServer: TAKEOFF;0.000000
ParrotTest: source sending message
ParrotTest: exiting
send@nodeServer: LAND;0.000000
init@commServer: listening on 10.0.0.3
message@nodeServer: (TAKEOFF, 0.000000)
basic: broadcast message
message@commServer: sending go! from interface
with address 10.0.0.3
basic: exiting
message@commServer: go! from 10.0.0.3
message@commServer: dropped
message@nodeServer: (LAND, 0.000000)
exit@nodeServer: shutting down
exit@commServer: shutting down
```

```
Sample AT commands:
AT*REF=21625,290717696<CR>
AT*PCMD_MAG=21626,1,0,0,0,0,0,0,0<CR>
```

```
init@nodeServer: listening on socket ./parrot.sock
send@nodeServer: TAKEOFF;0.000000
init@commServer: listening on 10.0.0.1
ParrotTest: sink waiting for message
message@nodeServer: (TAKEOFF, 0.000000)
message@commServer: go! from 10.0.0.3
basic: rec'd message
ParrotTest: received message
send@nodeServer: FRONT;0.500000
move@nodeServer: duration 2 seconds
message@nodeServer: (FRONT, 0.500000)
basic: rec'd message
send@nodeServer: STOP;0.000000
send@nodeServer: CLOCKWISE;0.500000
move@nodeServer: duration 3 seconds
message@nodeServer: (STOP, 0.000000)
message@nodeServer: (CLOCKWISE, 0.500000)
ParrotTest: exiting
send@nodeServer: STOP;0.000000
send@nodeServer: LAND;0.000000
message@nodeServer: (STOP, 0.000000)
message@nodeServer: (LAND, 0.000000)
exit@nodeServer: shutting down
basic: exiting
exit@commServer: shutting down
```

Collision Avoidance

Problem Division



Problem Division

Deployment: Problems

Wider Uses - Why Drones?

Asking the question “where are drones better than people?”

- Places where it is unsafe for humans.
- Situations where it is cheaper for unmanned work.
- When needing a view from the sky.

Wider Uses - Firefighting

80-90% of area burned in California is caused by 1% of the largest fires [4].

A network of drones - searching and waiting signs of a forest fire.

Alert the wardens so it can be contained.

Could also be used to airdrop new static nodes for a conventional sensor network.

Wider Uses - Mapping

Many places too dangerous for people to go [5]:

- Abandoned Mines
- Unstable Buildings
- Areas of high radiation or toxicity
- Space…

Each drone maps a different area. Information passed back to base station for integration into a complete map.

Outreach Implications

The 2015 RCUK Public Engagement with Research Strategy [2] sets out three main areas of research engagement:

- Involving the public with the research process
- Enabling public views to inform research strategies
- Engaging and inspiring the next generation

We believe that octoDrone is an excellent tool for academic outreach. We have created a platform which is powerful and extensible, whilst keeping it accessible to students.

Involving the Public with the Research Process

- Open access does not help with public engagement [3]
- We need to be creating materials written for a non-technical audience
- Visualisation allows for effective transfer of ideas
- The internet allows us to create materials for citizens to use in their own time, at their own pace

BIOMEDICAL WARWICK ENGINEERING OUTREACH

SWE MITSWEOutreach

Oxford Materials Outreach

Open Acces logo art designer at PLoS, modified by Wikipedia users Nina, Beao, and JakobVoss

Enabling Public Views to Inform Research Strategies

BBC (2016) Drone complaints to Welsh police forces rise. Published 05/05/2016.

- Research strategies should be driven by the issues facing society
- The public need to know about a topic in order to have an opinion about it [3]
- This can primarily be achieved through outreach tools such as ours
- Familiarity helps to counteract negative media presentation

Engaging and Inspiring the Next Generation

Inspiring the next researchers

- Summer schools and workshops are ideal places to inspire
- These environments are often hands on
- Increases the scope of problem solving from one craft to many

Making citizens more informed

- Fostering a better understanding of the technologies that are shaping our world
- Giving people an opportunity to better reason about the decisions which will affect their lives

Further Work

Performance Improvements

- Broadcasting is O (|*messageables*|) but could be modelled as a graph
- Drone upkeep is O(|*drones*|) but could be O(1) if threaded
- Tailor the number of software threads to the number of hardware threads

Content Improvements

- Add implementations of existing algorithms
- Add libraries for deployment to additional hardware
- Add modifications to allow for OSI level 2 protocols

References

[1] John Paulin Hansen, Alexandre Alapetite, I. Scott MacKenzie, and Emilie Møllenbach. The use of gaze to control drones. In Proceedings of the Symposium on Eye Tracking Research and Applications, ETRA ’14, pages 27-34, New York, NY, USA, 2014. ACM.

[2] Research Councils UK, (n.d.). RCUK Public Engagement with Research Strategy. [online] Available at: http://www.rcuk.ac.uk/documents/publications/rcukperstrategy-pdf/ [Accessed 4 May 2016].

[3] Hone, D. (2013). Why should academics get involved in outreach?. The Guardian. [online] Available at: https://www.theguardian.com/science/lost-worlds/2013/jan/02/dinosaurs-fossils [Accessed 4 May 2016].

[4] Stephen J Pyne et al. *Fire in America. A cultural history of wildland and rural fire.* Princeton University Press, 1982

[5] Gem Systems Advanced Magnetometers. Uavs - pathway to the future - gem systems, 2015

Questions