octoDrone: Developing Heterogenous Drone Platforms

{ Alex Henson, Ben De Ivey, Jon Gibson, and William Seymour } Computer Science, University of Warwick

Drone Networks

Drones present a powerful way of extending traditional sensor networks, and are currently available at low cost for recreational and commercial use. By allowing for sampling of the environment at any point in 3D space, quadcopters can be more efficient and reduce costs compared to existing systems.

A Parrot AR 2.0 quadcopter over the Nevada desert [1]

Current interesting uses for drone networks being researched include Disaster recovery after earthquakes and flooding, as well as the monitoring of active volcanoes. Most drone networks are heterogeneous: they feature many different models of drones from different vendors, posing a challenge when it comes to creating programs.

Objectives

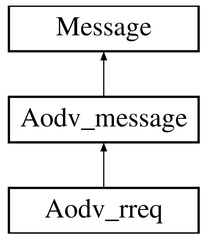
The aim of the project was to create a platform for the simulation and deployment of drone-based sensor networks. We aimed to develop a generic solution to the problem of modelling airborne sensor networks, including those comprised of heterogeneous hardware.

We aimed to demonstrate the effectiveness of the simulator by establishing a framework for running simulations on real hardware, without requiring users to change their program code. We hope that this solution will provide a basis for teaching, and research into the way that these networks are used in the future.

Simulation

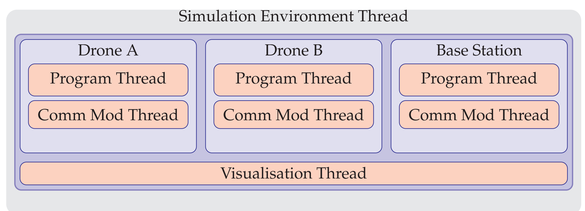
We designed our simulator, octoDrone, to be accessible, extensible, and lightweight. By simulating solutions for drone networks, one can test designs at scale (in order to ensure correctness). This approach also allows for defects to be discovered before deployment to hundreds or thousands of drones.

Accessibility was achieved by exposing a limited set of API commands which still allowed for the creation of powerful programs. More complicated actions, like routing messages between drones, are handled by separate modules to promote simplicity whilst still allowing for low level control.



In this way, developers can extend existing classes to create their own application specific instances. The AODV RREQ class is a good example of this, extending the AODV message class which itself inherits from a basic message class.

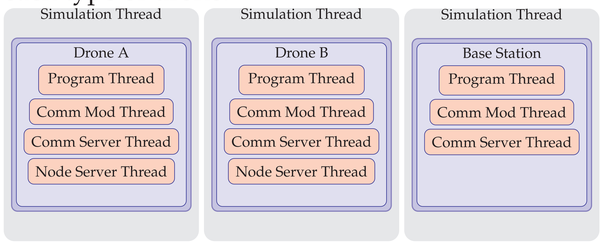
Each drone has separate threads representing its control and communication code. On top of this, the simulation and environment and visualisation layer have additional threads.



Communications modules allow for interrupt driven and time driven messaging. Inbound and outbound message queues are used to keep track of messages waiting to be processed, and the actual passing of messages (after processing) is handled by the simulation environment.

Deployment

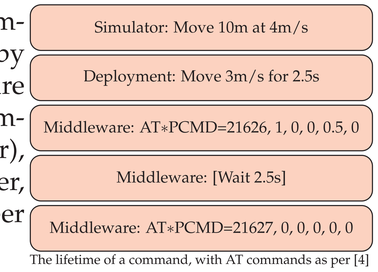
This part of the project aimed to enable the running of octoDrone simulations on real hardware. We adapted the simulator so that sharded instances could be run on each quadcopter. This prevented the need to rewrite correct programs for each type of drone.



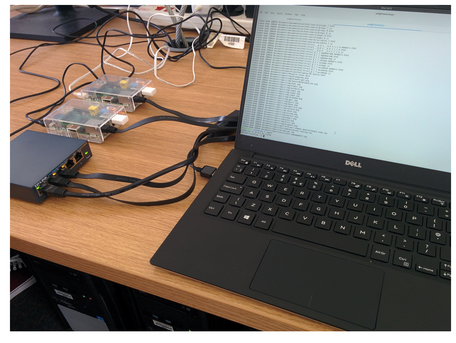
Because the Parrot AR 2.0 drones used in the project could not be programmed directly, each was paired with a Raspberry Pi running octoDrone. These units were connected via Ethernet, and communicated with the drones via WiFi.

Messages are sent and received by a communication server thread. An additional piece of middleware written in Node JS (The Node server) [3] is required to translate simulator instructions for a drone into commands that the firmware can interpret (called AT commands).

These AT commands are sent by the middleware and include a command (land, hover), sequence number, and some number of arguments.



The lifetime of a command, with AT commands as per [4]



We were able to successfully demonstrate the deployment capabilities of octoDrone, and show that this approach can be used to make any commercial quadcopter autonomous.

Use Cases

Research

Network simulators form the basis for a large amount of research into sensor networks and IoT devices. By allowing researchers to test their work under real world conditions, more representative results can be obtained, leading to better research.

Additionally, octoDrone is flexible enough to allow for almost any conceivable feature to be implemented with minimal effort. This modular approach is vital when research requirements are changing rapidly in this evolving field.

Outreach

With sensor networks, demonstrating research to students and members of the public can be difficult due to the many layered mechanics involved. Simulation offers a visual means of presenting this information, making it easier to understand.

Being able to deploy programs to drones opens up opportunities in schools, where hands on time with hardware is at a premium. Being able to see a physical result for their work is a great motivator for students.

References

[1] Parrot SA, 2009. Promotional Materials. [online] Available at <http://ardrone2.parrot.com/ardrone-2/specifications/>.
[2] Mottola, L., et al., 2014. Team-level programming of drone sensor networks. 12th ACM Conference on Embedded Network Sensor Systems.
[3] Felix GeisendÃűrfer, 2014. Node JS ar-drone package. [online] Available at <https://github.com/felixge/node-ar-drone>
[4] Stephane Piskorski, Nicolas Brulez, Pierre Eline, and Frederic D'Haeyer, 2012. Parrot SA AR Drone Developer Guide.