Self-Driving System for UGV

Yan Cao, Chemistry PhD at UCLA

Yucheng Yao, Mathematics of Computation Major at UCLA

01 ABOUT US

OUR AMBITION

Imagine waking up and being served hot breakfast by no one else but your trusted Roomba. Imagine never having to load the dishes again. Imagine the amount of liberty freedom you will finally enjoy if you never need to do household chores again.

This is what we try to achieve here at flamepunk. We are building a control system based on multihop wireless communication for unmanned ground vehicle cluster.

MEET FLAMEPUNK

YUCHEN YAO

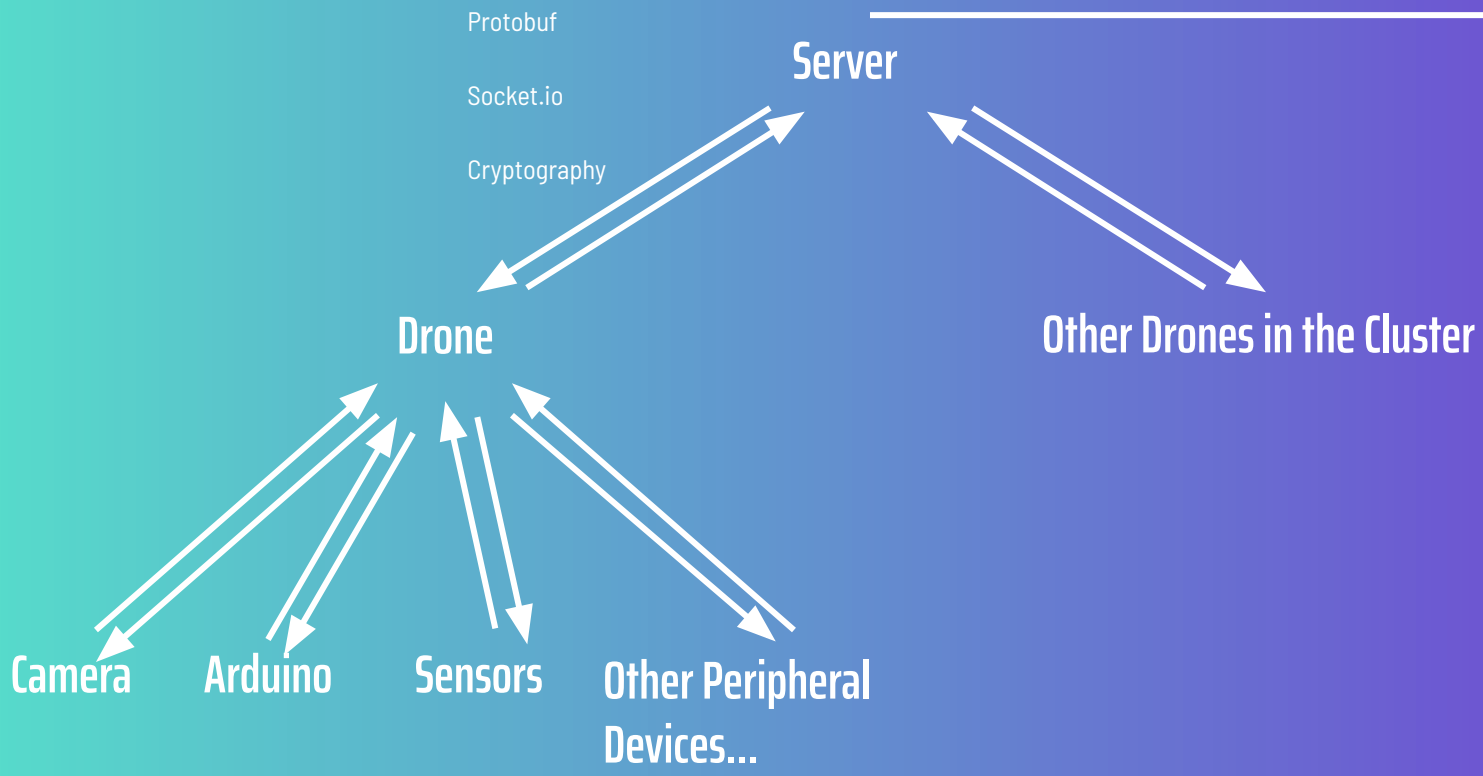
Mathematics of Computation Major at UCLA

YAN CAO

Chemistry PhD at UCLA

CORE FEATURES

GENERAL DESIGN



CORE FEATURES

EXTENSIVITY

Besides the head node, this doesn't require any additional hardware support other than Wifi module or Bluetooth. Any device conform to the system's communication standard can be added to the network without too much overhead

PRIVACY

This system is different from current IoT devices in that it only has a single access point to the Internet, the head node. The node has much more processing power than the devices that it controls, adding another layer of protection.

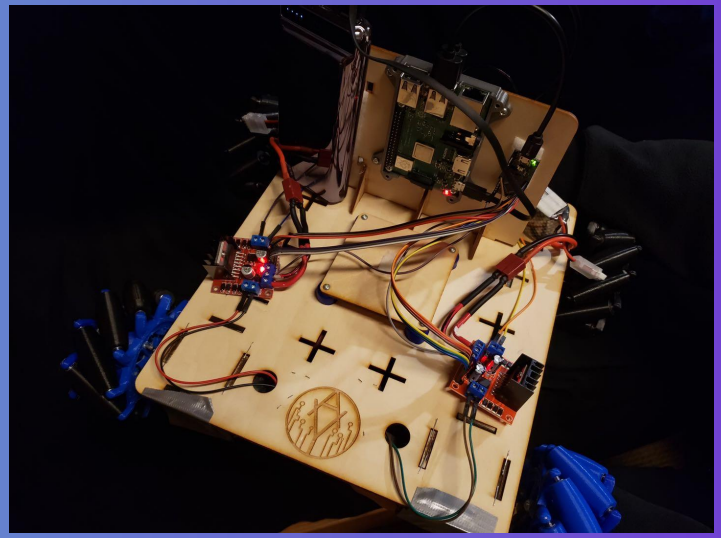
CURRENT PROGRESS

02

CURRENT PROGRESS

At the current state, we achieved bidirectional communication between the server, my laptop, Raspberry Pi, the head node, and Arduino. We can now send commands from a terminal on the PC side to control movements of the UGV and read from an onboard GPS module.

We finished the overall design of the system and set goals for the first iteration (we are currently at around 20%). This is the basic model for the system.



03 CHALLENGES

CHALLENGES

System Setup

The major problems that we encountered were setup issues. Making the Raspberry Pi an access point alone took 2+ hours because it is running an ISO distributed by Ubiquity Robotics and many services were running.

We eventually made it work

Time

Moreover, due to the lack of personnel, we had to abandon the original design.

Instead of creating a node.js server and Flutter front end, we fell back to pure C++ client and server model.

Cluster Control Design

Dynamically connect to multiple drones and coordinate them according to their functionality.

We are still working on the designing.

FIRST ITERATION 04

FIRST ITERATION

use Socket.io for communication between Pi and server so that we can have a Flutter or React front end

integrate protobuf to the system and incorporate ROS framework (Robotics Operating System)

utilize on board sensors -a GPS module, a 9-Axis Attitude Gyro Accelerometer Compass Magnetic Field Sensor, and cameras

05 SECOND ITERATION

SECOND ITERATION

rely mainly on machine vision to calculate depth of field, which will then be converted to a 3D map

add asymmetric cryptography for communication between Pi and server

redesign hardware and add a sophisticated power management system

06 FUTURE PLANS

FUTURE PLANS

More Accessories

We plan to add accessories such as robotic arms

More Personnel

to improve team efficiency

More General Control Cluster of Drone

being able to utilize multiple kinds of unmanned vehicles

THANKS

Please send your questions to
ausaryao00@gmail.com