**Graduate Projects**

University of Colorado at Boulder

Aerospace Engineering Sciences

ASEN 5018/6028 –Spring 2015

|  |
| --- |
| **Drones Versus Zombies (DVZ)**  **Communication Subsystem (CommS)**  **Summary/Continuity Document** |

|  |  |  |
| --- | --- | --- |
| **Approvers List** | | |
|  | Title | Name |
| Prepared By | CommS Lead | Quinn McGehan |
|  |  |  |
| Approved By | CommS Lead | Quinn McGehan |
| Approved By | Systems Engineer | Edward Scott |
| Approved By | Project Manager | Garrett Brown |

# Introduction & Summary

The communication subsystem is responsible for the setup and configuration of the necessary system components, as well as the transfer of data between various components of the project. Specifically, this entails configuring the ODROID computers, as well as the ground station with the correct operating systems, and ensuring the programs running on the on-board computers on the drones can communicate with the programs running on the ground station computer. Further, this subsystem is responsible for the design of the application layer graphical user interface (GUI) in accordance with requirement SWE\_010.

Currently, the communications subsystem has met all networking requirements for this semester. This includes the selection of the Robot Operating System (ROS) for communication between hardware components and software, as well as between various software packages across the entire system. The initial ROS architecture designs for the sensing system, as well as the detection system, are included in Figures 1 and 2.

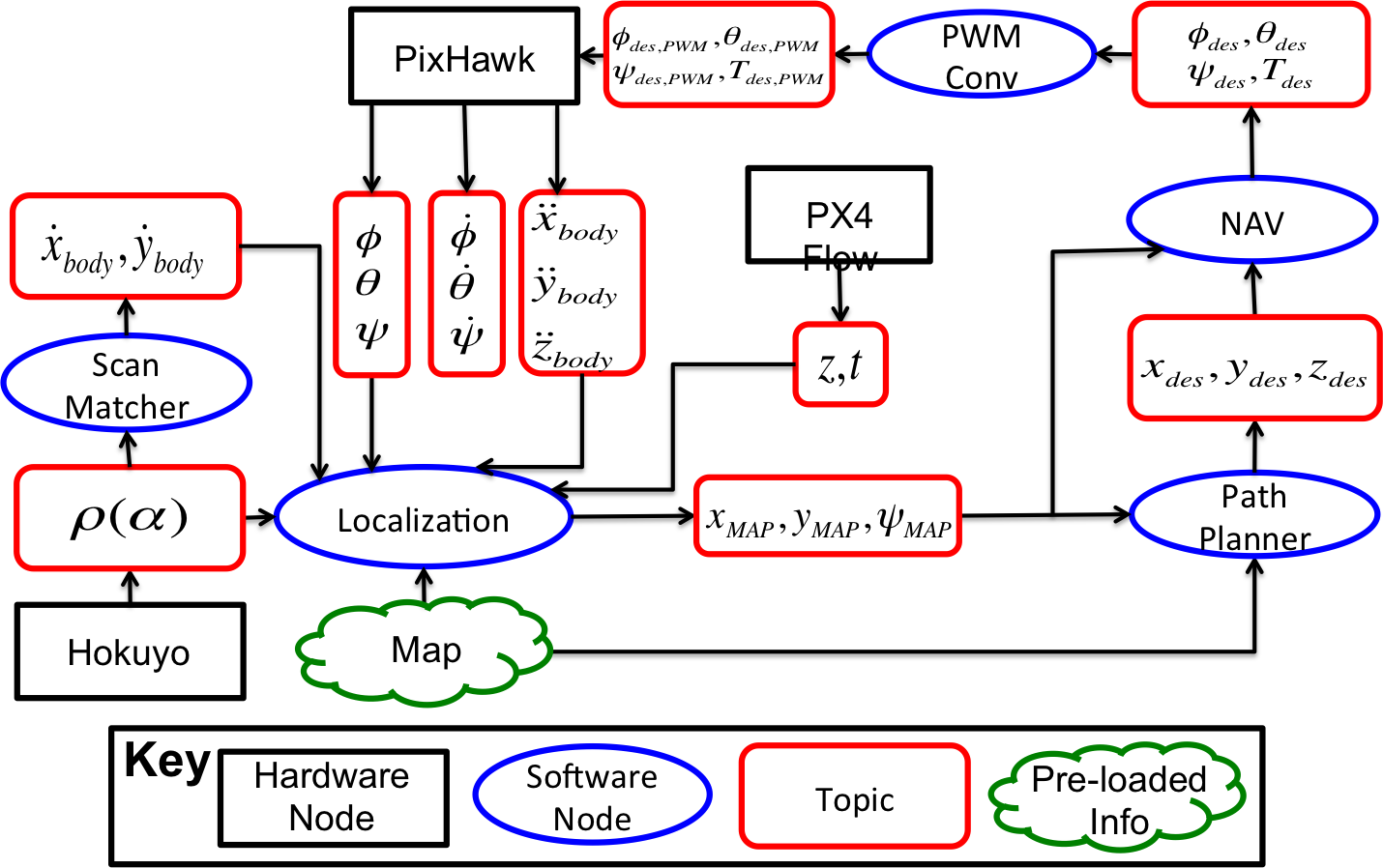


Figure : Sensing ROS Architecture

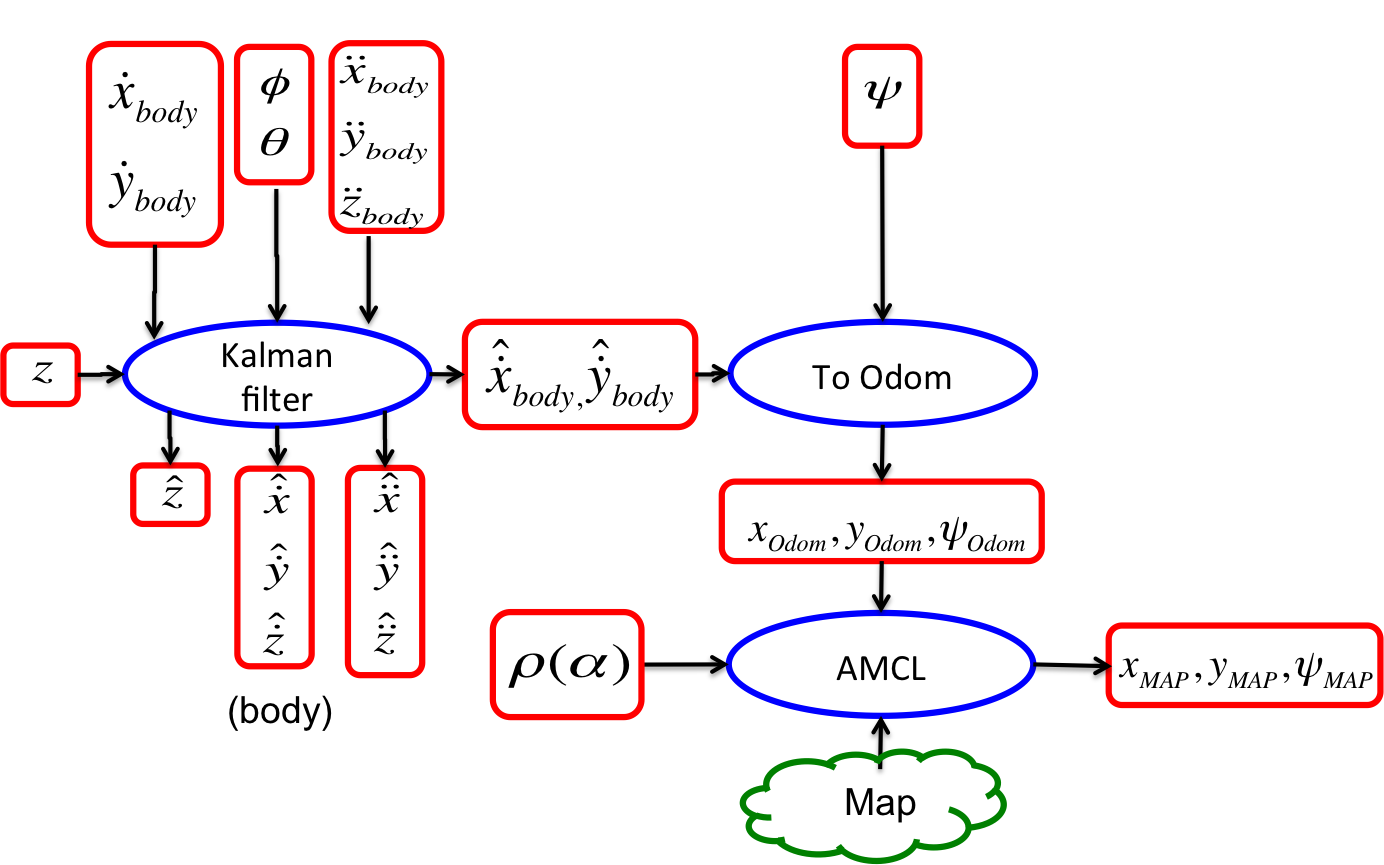


Figure : Localization Architecture

At a lower level, the communications subsystem has chosen to leverage the available WIFI networks in RECUV (while in Fleming), as well as the provided university network (UCB Wireless), as the means of communication between the drones and the ground station.

# Semester Report

## Objectives and Tasks List

The objectives for the Spring of 2015 did not drastically change from the fall, but focused more on maintenance and improvement of the architecture from the prior semester. Due to the continually altering architecture, and the descope of the detection requirements, the GUI was decscoped for this semester. Further an accurate simulation of the system was looked into this semester. This was to be implemented in Gazebo, which would have allowed easy plug and play between simulation and physical environments

Completed:

1. Recontacted OIT in order to change MAC address from broken WIFI dongle to new WIFI dongle
2. Determined method for implementing simulation in Gazebo.

Incomplete:

1. Implementing simulation in Gazebo.

## Issues

1. The initial wifi dongle for Wolverine was broken during testing, so it was necessary to get a new one running. The need for a new wifi dongle required recontacting OIT to change the MAC address of Wolverine from the old dongle to the new dongle
2. Some latency issues were observed that may or may not have been related to the wifi. All diagnostics that were run seemed to indicate that wifi was not the cause, but performance in the Lockheed Martin Hallway was in every case worse than performance in recuv even using the RECUV\_VICON router.
3. The clock on the ODROID resets to Jan 1, 2015 every time it is power cycled, which led to issues when using a network solution that did not connect to Internet (using point-to-point instead).

## Lessons Learned

1. Setting up the 2nd ODROID (Magneto) to match the 1st (Wolverine) was nontrivial. The initial thought was that replicating the system between ODROIDS would be as simple as loading the same packages, however, the actual replication of the ODROID proved more complex. Solutions to this problem are discussed in Section 2.4
2. Debugging is not an easy task. Simply because there are latency issues does not mean it is only the network connection. For instance some issues were caused by ROS subscription rates as opposed to the wifi connection.
3. Networks (even small ones) are difficult to maintain and keep functional. I have a new profound respect for those in the IT business.

## Procedures

This section contains the procedures from last semester, as well as new updates intermixed as necessary.

### ODROID

ODROIDs come pre-loaded with Lubuntu 14.04 on the eMMC module. The size of the eMMC modules is 16 GB. Initially, it was decided to use Lubuntu 12.04 on the ODROIDs, however, efforts to find and install a stable build of Lubuntu 12.04 were unsuccessful. Thus, it was decided to return to Lubuntu 14.04. Prior to installing the OS on the ODROID, ensure the eMMC module is partitioned correctly. To partition the ODROID follow the steps in the following document:

* /DVZ/Software Element/Documentation.
* README\_partition\_odroid

Currently, the Lubuntu 14.04 image that is installed on the ODROIDs is located on the ground station computer (Cerebro) in:

* ~/Documents/odroid\_img

In order to install the image on the ODROIDs follow the steps located in the following document:

* /DVZ/Software Element/Documentation/
  + README\_configure\_odroid

The final step in the initial configuration of the ODROID is installing ROS. The ODROIDs should be installed with ROS Indigo. Because ODROIDs have ARM processors, it is important to follow the installation directions on the ROS wiki for INSTALLING ON ARM PROCESSORS. These instructions can be found at the following link:

* <http://wiki.ros.org/indigo/Installation/UbuntuARM>

Following the installation of ROS, the packages necessary to run DVZ need to be installed. As noted in the issues section this was a nontrivial process, and has been documented in:

* /DVZ/Software Element/Documentation/
* README\_install\_packages

### Ground Station:

The ground station, nicknamed Cerebro, was configured with Ubuntu 14.04 LTS, and SHOULD NOT BE CHANGED. If ROS happens to not be installed on Cerebro, follow the instructions found at the following link:

* <http://wiki.ros.org/indigo/Installation/Ubuntu>

For specific ROS package installation and use, see the appropriate subsystem continuity document.

### Communicating over RECUV\_VICON

The platforms were configured to communicate over the RECUV\_VICON WIFI network by assigning each individual platform a static IP address on the RECUV\_VICON wireless router. To set up a static IP on RECUV\_VICON see the instructions in the following document:

* /DVZ/Software Element/Documentation.
* README\_recuv\_vicon

NOTE: At this time all static IPs on RECUV\_VICON for DVZ should be assigned in the range to:

* 192.168.20.40 to 192.168.20.49

The current static IPs being utilized by DVZ can be found included in Table 2.1:

NOTE: The MAC addresses for the drone platforms (i.e. Wolverine and Magneto) are dependent on the WIFI module that they are connected to. Thus, each ODROID should only be used with the corresponding WIFI module. Currently, the WIFI modules and ODROIDs are labeled with the particular platform they correspond to.

This semester the original wifi dongle for Wolverine was broken in an uncontrolled landing. This required the MAC addresses associated with the DNS and the specific static IPs associated with Wolverine to be set to the MAC address of the new wifi dongle. Table 2.1 reflects this update.

Table 2.1: Static IP addresses for RECUV\_VICON Network

|  |  |  |  |
| --- | --- | --- | --- |
| Device | Platform | MAC Address | IP Address |
| Cerebro | Ground Station | D0:7E:35:07:D4:B0 | 192.168.20.40 |
| Wolverine | Quad | 7C:DD:90:52:13:98 | 192.168.20.41 |
| Magneto | Quad | 7C:DD:90:52:1A:AD | 192.168.20.42 |

### UCB Wireless

An overarching goal of the DVZ project is to leverage and combine as many existing components as possible. Because the goal is to play the DVZ game within the Engineering Center, it would be convenient and useful to leverage the existing university WIFI network. Unfortunately, communicating between platforms on UCB\_Wireless is not as simple as it is on the RECUV\_VICON network. Similar to RECUV\_VICON, in order to communicate between platforms while on UCB\_Wireless, static IP addresses are required. Static IP addresses were requested from the OIT department, under case number INC0134678, under the purview of Nicholas Miller. The OIT department provided static IP addresses for DVZ within the Engineering Center. According to OIT, static IPs can only be provided for one building at a time. For the devices to be assigned their static IP addresses while in the Engineering Center, they must connect to the SSID (or network name) UCB\_Fixed. The static IP addresses for the platforms in the Engineering Center can be found in Table 2.2.

NOTE: OIT requested that they be informed at the end of this iteration of the project so that they can reclaim the IP addresses.

As described above the wifi dongle for Wolverine was changed during the Spring 2015 semester, so the MAC address for its IP address on UCB wireless also had to be updated to reflect the change. OIT was contacted again and changed the old address to the new address. This update is reflected in Table 2.2.

Table 2.2: UCB\_Wireless Static IP Addresses

|  |  |  |  |
| --- | --- | --- | --- |
| Device | MAC Address | IP Address | DNS |
| Cerebro | D0:7E:35:07:D4:B0 | 10.201.0.12 | Engr2-0-12-fixed |
| Wolverine | 7C:DD:90:52:13:98 | 10.201.0.10 | Engr2-0-10-fixed |
| Magneto | 7C:DD:90:52:1A:AD | 10.201.0.11 | Engr2-0-11-fixed |

A major issue encountered when switching between Internet SSIDs (i.e. RECUV\_VICON and UCB\_FIXED) is that the IP addresses differ from one another. This issue was resolved by setting the hostname of the ODROID in the **/etc/hosts** file to the DNS name set by OIT (we did not really have a choice in the names they provided). This change allows the ROS master to be enabled independently of the ground stations actual IP address (At least in the Lockheed Martin and on RECUV\_VICON).

Finally, in order to determine the quality of the connectivity in the Lockheed Martin Hallway a wifi signal heat map was generated which depicts the connection strength. This was done using the floor plan and a program from the Internet called HeatMapper. While the fidelity of the maps generated has not been verified by other means, they do seem to indicate that the area of the hallway that was mainly used for testing (the far left horseshoe) had poor-to-medium signal strength. This is potentially responsible for the latency issues, and thus a reason to rethink the use of UCB\_Wireless. The heat maps can be found in:

* /DVZ/Verification\ Validation\ Test/Lockheed\ Martin\ Hallway\ Heat\ Maps/

### Initializing ROS Master

Once the computers are configured properly and are configured to properly communicate across a WIFI network, the ROS master must be started and enabled on all platforms in order to properly communicate between them. Because the goal is to have multiple drones communicating with the ground station, it was decided to make Cerebro the ROS Master. The instructions for starting and enabling Cerebro as the ROS Master can be found in the following document:

* /DVZ/Software Element/Documentation.
* README\_SETUP\_MASTER

### Statistics monitoring

In order to troubleshoot the latency issues that were experienced when running the full system architecture many different methods were used to determine the data flow at different points in the system. In order to graphically depict the overall ROS architecture, with summary statistics included prior to running roscore execute:

* >> rosparam set enable\_statistics true

then execute roscore and start packages as usual. To pull up the architecture picture execute:

* >>rqt\_graph

A GUI with a diagram such as in Figure 2.1 should come up.

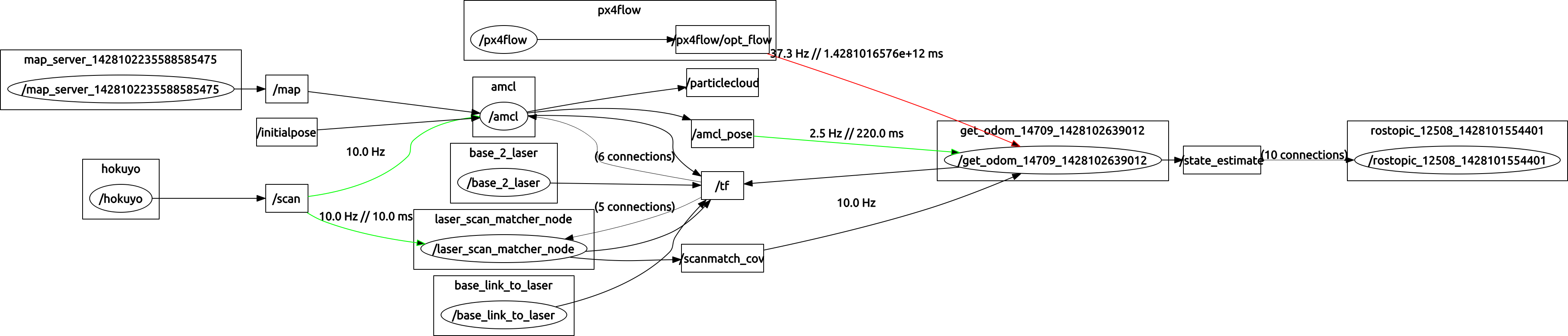


Figure .1 Example rqt\_graph diagram

The boxes and circles represent the nodes and topics that are currently running. The lines between the boxes indicate the publish/subscribe structure. With the statistics enabled the line colors represent the amount of latency in that path, the line thickness represents the amount of data flowing along that path, and the numbers indicate the subscription rate, and latency in (ms). For more information on rqt\_graph see the rqt\_graph wiki at:

* <http://wiki.ros.org/rqt_graph>

While these statistics are helpful for understanding the flow of data within the ROS framework it does not provide insight to the traffic over the network (e.g. wifi). This was done by installing **bmon**, a statistics monitoring program onto the odroids. This was done using:

* >>sudo apt-get install bmon

Once it is installed execute:

* >> bmon

This will bring up information about the etho and wlan connections on the device. This is useful for checking how much data is being passed over the wifi network when the system is running. Note: This can be run from either the ground station or the odroids.

# Next Semester/Future Expectations

## Prioritized List of Tasks and Objectives

1. Re-think the use of wifi, more specifically UCB\_Wireless, for communication as it definitely seemed to cause issues in the system.
2. Get Gazebo simulation up and running (However this should maybe be its own subsystem in the future depending on the scope and value to the project)
3. Create GUI

## Starting Points

1. Look into other options for communicating without UCB\_Wireless. This could mean a wifi hotspot near the ground station, a separate router (for instance the RECUV\_VICON setup). Also talk to James about his methods of communication in his work.
2. This semester it was determined where and the Gazebo simulation would need to be modified in order to use it for simulating the DVZ system, so implementing those changes would be the place to begin with that. Although this may not be as easy as it sounds, so I recommend numerous people working on it.
3. Reconsider the options for a GUI. Potentially MATLAB based, or web based. If a web based option is chosen talk with Nick Sweet, as he has experience in this area.

## Improvement, Updates, Verification

1. Improving the initialization of ROS to make it completely independent of IP or DNS name would be convenient as this issue made life difficult numerous times.
2. Write a script, or scripts, that start the necessary packages and programs for the system to run. Currently setting the system up consists of numerous commands being run in numerous terminals, which then requires ssh’ing into the ODROID quite a bit, which is a time sink. The system would really be streamlined with a script that started everything needed from one terminal.