Swarming Robots

May 1308

Douglas Feldmann
Matt Henderson
Matt Klinkefus
Zach Njus

Table of Contents

Overall Platform Setup 5

Option 1 5

Option 2 5

Option 3 6

Conclusion 6

Development Platform 6

Operating System 10

Supplement Microcontroller 11

RC Car 12

Brushed DC 12

Brushless DC 12

Type 12

Scale 12

Conclusion 13

Object Detection 13

GPS Unit 14

Options 15

Conclusion 15

Directional Unit 16

Options 16

Conclusion 16

Imaging Unit 16

RC Car Hardware Implementation 18

Communication 18
ZigBee 18
WIFI 19
Frequency Band 20
Data Interchange Format 20
Transmission and Response Rates 20
Conclusion 21
Video Streaming 21
GStreamer 21
Custom Streamer 21
Conclusion 21
Command Center Design 22
Platform 22
Android 22
iPhone 22
Netbook 23
Comparison 23
Conclusion 24
Netbook Selection 24
Architecture 24
Communication 24
Command Center Interface 25
Reference Documents 27

List of Figures

Figure 1: PandaBoard 10
Figure 2: Maple Mini 11
Figure 3: Arduino Uno 11
Figure 4: Exceed-RC Infinitive EP Electric Truck 13
Figure 5: Venus GPS with SMA Connector 15
Figure 6: HMC6352 16
Figure 7: Logitech Webcam Pro 9000 17
Figure 8: System Diagram of RC Car Hardware 18
Figure 9: Mission Priority Screen 24
Figure 10: Swarm Screen 25
Figure 11: Logitech Webcam Pro 9000 25

List of Tables

Table 1: Comparison of Development Platforms 8
Table 2: Comparison of GPS Modules 15
Table 3: Comparison of Magneto Sensors 16
Table 4: Different Video Transfer Rates 19
Table 5: 802.11 Network Standards 19
Table 6: Comparison of Mobile Platforms 23

Overall Platform Setup

The different system setups that were considered for the project include using a single board computer, a phone, or a micro controller and router combination as the main devices on the car. These three options need to allow for either video processing or just relaying the video feed. Each setup would also include the following modules with it:

- Servos for aiming the camera
- Speed controllers for brush-less DC motors
- Servo for control of the steering mechanism
- GPS for location information
- Electronic compass to help compensate for incorrect GPS readings and bearing while at a standstill.
- Wireless module
- Main computing system e.g. phone, single board computer, microprocessor, etc.
- Camera for streaming video, and still photos
- Sensors for object detection and avoidance

Option 1

Option one is to use an Android phone as the main computing platform on the car, then using the IOIO or similar development platform create an interface for all other hardware to be controlled from the phone

- Pros
    - The phone would include a wireless communication module, camera, GPS, and a more than capable processor.
    - The phone's power system could be separate from the motors' power system, allowing for easier integration
- Cons
    - This setup would have two separate power systems, which could be a hassle for recharging and other maintenance
    - The phone would have to be rotated when the operator wants to control the camera, which could damage control lines for propulsion system
    - The phone's screen and many other parts would be wasted, and therefore add unneeded weight.

Option 2

Option two is to use a single board as the main computing platform for the RC car. This would allow for enough processing power to handle streaming video feeds. A microprocessor would then be used for all of the low level control of the car such as motors. This approach would allow for more creativity because the project would not be limited by what the IOIO board would give access to, but would require a greater effort for implementation.

- Pros
    - Could allow for better implementation of all devices, since it will not depend on what can be done from a phone, and would have full access to all unused processor I/O on the board
    - Less wasted space and weight
    - A single board computer is less expensive than buying a cell phone without a contract
    - Only one power system to worry about
- Cons
    - More implementation overhead, more time

Option 3

Option three is to use a microprocessor as the main computation unit. This would mean that the microcontroller would not be able to process the video feeds. If this option were used, an IP based camera system would need to be used, and a router would need to be placed on the car. This would draw more power, consume more space, and be a less elegant and more awkward setup.

- Pros
    - A microprocessor would use much less power
    - A microprocessor would take up much less space
    - Setting up I/O for most microprocessors is fairly simple
- Cons
    - Less available computation power
    - A more expensive IP style camera would be needed
    - Even more overhead to implement wireless protocols
    - Less space for program memory

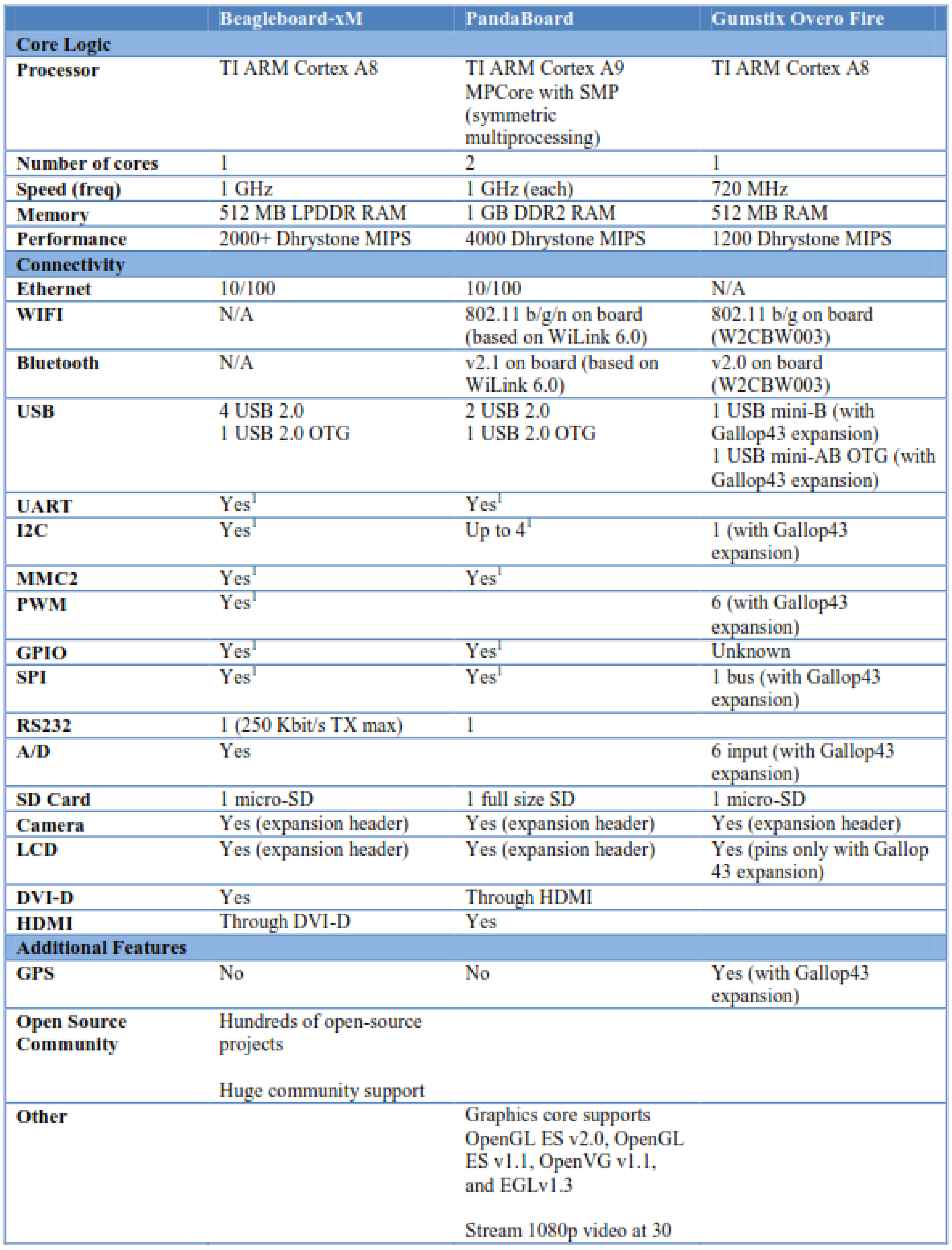
Conclusion

From reviewing the preceding three options that were considered, it is apparent that the second option would be the best suit for this project. It will allow for the necessary processing power for the video processing and for other components to be implemented easily. It will also add an expandability factor that could be helpful for further improving on the design.

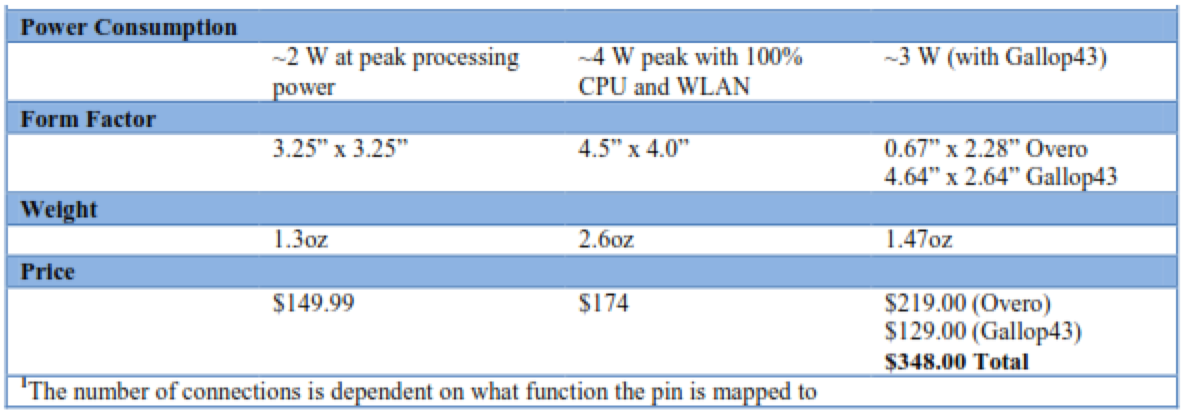
Development Platform

Since the project requires the capability to process video taken from a mounted USB webcam, other data from the RC car, and relay other RC car's data in an Ad-Hoc manner, it requires a processor capable for this workload. A simple Arduino would be ideal if the project did not have to support streaming video. Since Arduino does not come close to the processing power needed, a few high end embedded development platforms came into consideration.

To make a good justification of what development platform should be used, three different boards were compared: Gumstix, Beagleboard, and PandaBoard. Below is a table detailing the specifications of each board.



	Beagleboard-xM	PandaBoard	Gumstix Overo Fire
Core Logic			
Processor	TI ARM Cortex A8	TI ARM Cortex A9 MPCore with SMP (symmetric multiprocessing)	TI ARM Cortex A8
Number of cores	1	2	1
Speed (freq)	1 GHz	1 GHz (each)	720 MHz
Memory	512 MB LPDDR RAM	1 GB DDR2 RAM	512 MB RAM
Performance	2000+ Dhrystone MIPS	4000 Dhrystone MIPS	1200 Dhrystone MIPS
Connectivity			
Ethernet	10/100	10/100	N/A
WIFI	N/A	802.11 b/g/n on board (based on WiLink 6.0)	802.11 b/g on board (W2CBW003)
Bluetooth	N/A	v2.1 on board (based on WiLink 6.0)	v2.0 on board (W2CBW003)
USB	4 USB 2.0 1 USB 2.0 OTG	2 USB 2.0 1 USB 2.0 OTG	1 USB mini-B (with Gallop43 expansion) 1 USB mini-AB OTG (with Gallop43 expansion)
UART	Yes[1]	Yes[1]	
I2C	Yes[1]	Up to 4[1]	1 (with Gallop43 expansion)
MMC2	Yes[1]	Yes[1]	
PWM	Yes[1]		6 (with Gallop43 expansion)
GPIO	Yes[1]	Yes[1]	Unknown
SPI	Yes[1]	Yes[1]	1 bus (with Gallop43 expansion)
RS232	1 (250 Kbit/s TX max)	1	
A/D	Yes		6 input (with Gallop43 expansion)
SD Card	1 micro-SD	1 full size SD	1 micro-SD
Camera	Yes (expansion header)	Yes (expansion header)	Yes (expansion header)
LCD	Yes (expansion header)	Yes (expansion header)	Yes (pins only with Gallop 43 expansion)
DVI-D	Yes	Through HDMI	
HDMI	Through DVI-D	Yes	
Additional Features			
GPS	No	No	Yes (with Gallop43 expansion)
Open Source Community	Hundreds of open-source projects Huge community support		
Other		Graphics core supports OpenGL ES v2.0, OpenGL ES v1.1, OpenVG v1.1, and EGLv1.3 Stream 1080p video at 30	



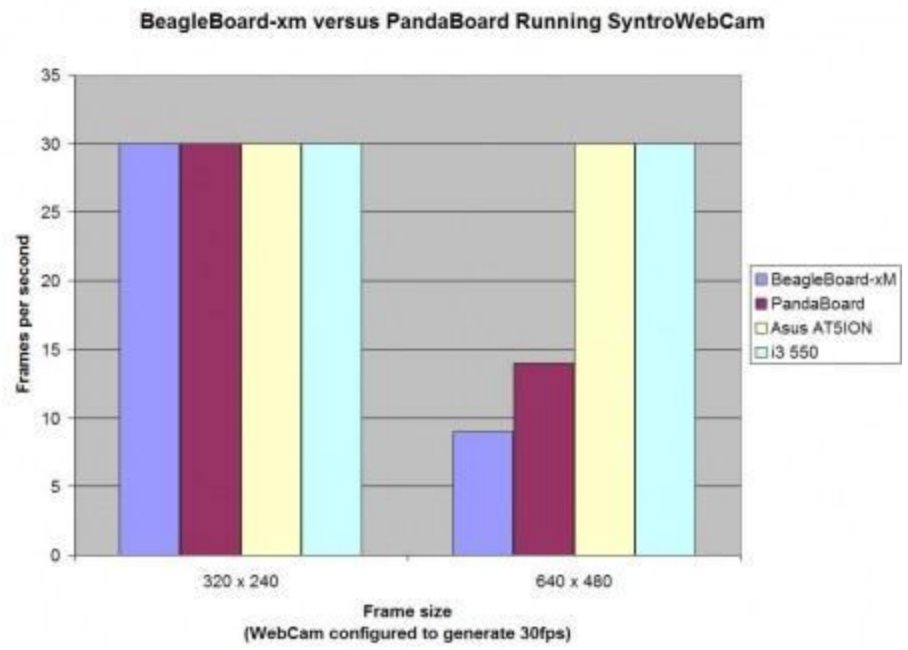
Power Consumption			
	~2 W at peak processing power	~4 W peak with 100% CPU and WLAN	~3 W (with Gallop43)
Form Factor			
	3.25" x 3.25"	4.5" x 4.0"	0.67" x 2.28" Overo 4.64" x 2.64" Gallop43
Weight			
	1.3oz	2.6oz	1.47oz
Price			
	$149.99	$174	$219.00 (Overo) $129.00 (Gallop43) **$348.00 Total**
[1]The number of connections is dependent on what function the pin is mapped to			

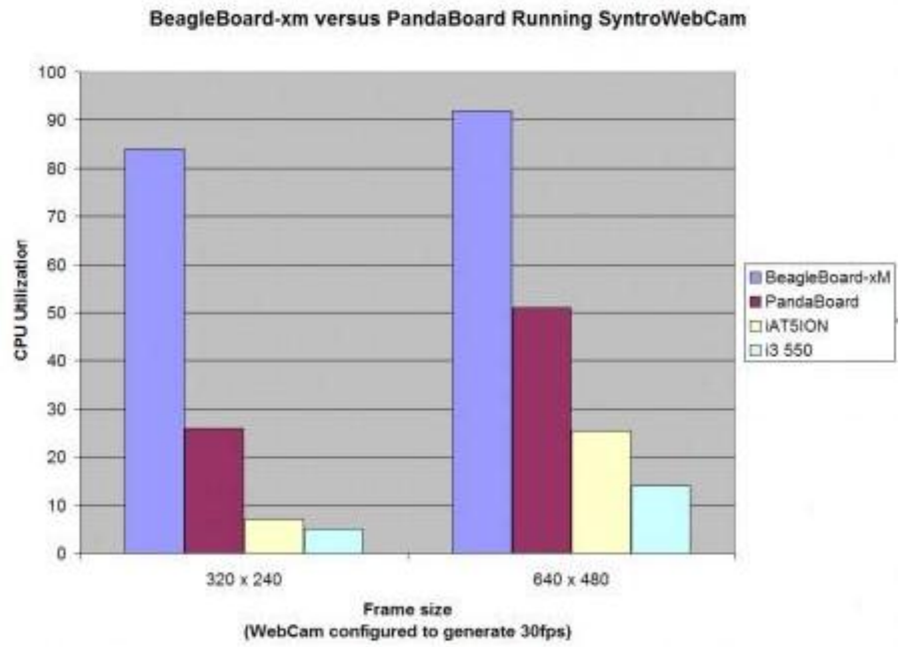
Table 1 – Comparison of Development Platform

By looking at this chart, the Gumstix Fire does not offer all the features as the PandaBoard and Beagleboard offer and its processing power does not match the other two. Since video streaming does take a lot of processing power, it is an important issue to consider when selecting a board.

The Beagleboard and PandaBoard are very similar. Both offer roughly the same functionality and capabilities and are roughly the same price. However, the PandaBoard offers an on-board 802.11n module which is the exact 802.11 protocol selected for use in this project.

In addition since it has been mentioned multiple times that video streaming is a must, the PandaBoard is the clear winner for processing power. Below are two graphs that address this issue.[1] They show the fps and percent CPU utilization of two different resolutions, 320 x 240 and 640 x 480, using SyntroWebCam. SyntroWebCam is a tool used by Syntro, a generic robotic operating system. All components communicate via TCP/IP so these graphs provide valuable insight into how well the vehicle will be able to send video wirelessly from the PandaBoard.





According to these graphs, the PandaBoard will be able to meet the video processing functional requirement laid out in the Project Plan with ample CPU power to perform other tasks.

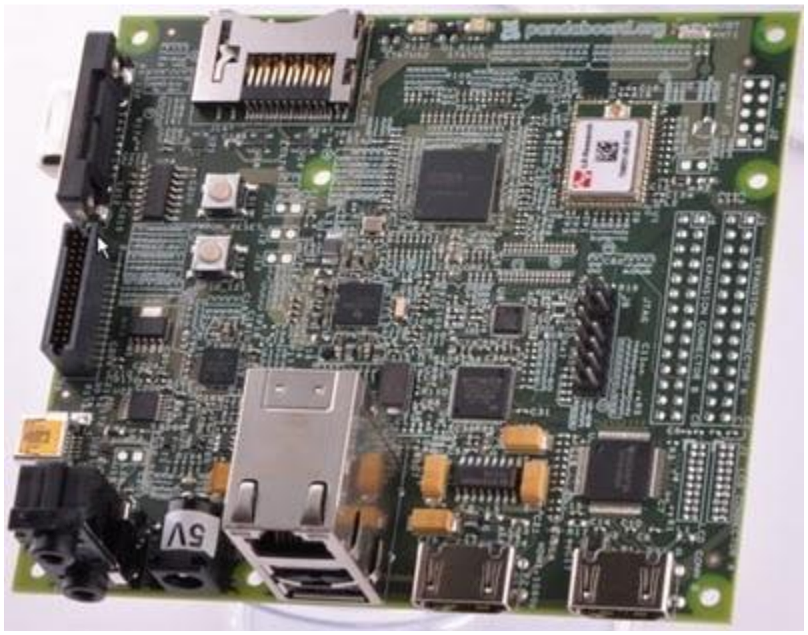


Figure 1 - PandaBoard

Operating System

This section will focus on what operating system is going to be used on the Panda Board. The options that are in consideration are Linux, Android, and Windows CE. The operating system needs to be a widely used OS so that there will be information widely available for reference. The OS would also be preferable if it was free of charge with an open source mentality so there is direct access to the source code if need be. These two preferences will rule out Windows CE.

Since the operating system will be running on the Panda Board, there will be a constrained amount of processing power available to consume. This means that a light OS with only the minimal requirements would be preferred so that unneeded tasks are not consuming resources. Android is an OS that is directed at consumer electronics that run on batteries. At the core, Android is based off of Linux with a slightly modified kernel. The major downfall of using Android would be many extra tasks running in the background in turn wasting energy.

The remaining option would be a distribution of Linux. There is a pre-compiled version of a minimal version of Linux built specifically for use with the Panda Board. This version would not have any unnecessary tasks running in the background and would allow for full integration of all modules needed for the project. This would be much more efficient than the other options.

From the arguments above, the minimal version of Linux built exclusively for the PandaBoard will be the best option for the OS.

Supplemental Microcontroller

Even though an Arduino microcontroller cannot perform the high-level processing as compared to the PandaBoard, it has two main features that are required for the project: PWM signal generation and reading and writing analog and digital signals.

To create a PWM signal, an interrupt within a microcontroller is triggered at a specified time defined by the microcontroller's timing counter. Current processes are suspended while the microcontroller creates a signal. The PandaBoard is capable of PWM signal generation but since it is dedicated for all high level programming that will utilize most of the CPU removing the overhead of suspending these processes for this simple function could harm overall performance of all functions.

The Arduino will be connected to the PandaBoard via SPI (Serial Peripheral Interface). This allows the PandaBoard to send commands and receive data from the Arduino. The rate of this data transfer will be determined in implementation.

The type of Arduino development board is still a topic of discussion. Two Arduino capable boards are the Arduino Uno and Maple-Mini.

Figure 2 - Maple Mini

Figure 3 - Arduino Uno

RC Car

There are three main types of RC cars on the market: gasoline, nitro, and electric. Due to their high noise levels, gasoline and nitro will not be pursued for this project. Even though electric is the only type left, there are two variations of electric RC cars that need to be compared: brushed and brushless.

Brushed DC

Brushed RC cars use electric motors with brushes in constant contact with a commutator to spin a permanent-magnet between two magnetic poles. These are very simple motors to design and operate by simply applying a positive voltage. However, their downfall comes to wasted power since the motor is always turned on, energy is lost due to the friction between the brushes and the commutator, and susceptible to mechanical wear between the brushes and the commutator.

Brushless DC

Brushless RC cars, as the name implies, use brushless DC motors. A brushless DC motor uses permanent magnets to rotate around a fixed armature. Since a brush and commutator are no longer used to turn the motor, an electronic controller is instead used to continually switch the phase of the windings in the magnets to keep the motor turning.

Brushless offers several advantages over brushed motors including more torque per weight, more torque per watt (increased efficiency), increased reliability due to no brush and commutator erosion, reduced noise, reduction of electromagnetic interference, no windings on the rotor reduce centrifugal forces, and the entire motor can be enclosed to protect it from foreign matter.

Type

There are many types of RC cars including but not limited to drift, rock crawler, on-road, off-road buggy, off-road monster truck, off-road truggy, off-road short course, and off-road ATV. For this project, off-road monster trucks will provide ample stability through tougher suspension, can handle more weight, and are able to drive across most outdoor terrain.

Scale

RC cars come in multiple scales that include but not limited to 1:16, 1:12, 1:10, 1:8, and 1:6. These scales do not necessarily mean each RC car at given scale are the same size. A scale shows the size of the model compared to the actual item it is supposed to represent. Therefore, a 1:8 RC model of a tank will be much larger than a 1:8 RC model of a sports car.

For this project, there needs to be enough room for implementation and testing. A 1:16 RC car will not provide ample room for implementation of all items needed. A typical 1:16 RC monster truck will be about 10” long by 8.5” wide. With the PandaBoard’s form factor, all peripherals, and all cabling needed will use up most of this area. In addition, all these items need to be located near the center of the RC car as much as possible to help maintain balance and reduce the effect to the RC car’s original center of gravity. Therefore 1:16 RC cars will be too small for the needs of the project and 1:10 scale with be pursued instead.

1:10 RC monster trucks generally have a length of 16” and a width of 11”. With this extra area, there should be ample room for implementation of all components while trying to preserve balance and center gravity.

Conclusion

With all these different options to weight, the best route for this project is to use a 1:10 Brushless RC Monster Truck. Below is a picture of a model that meets all these requirements.

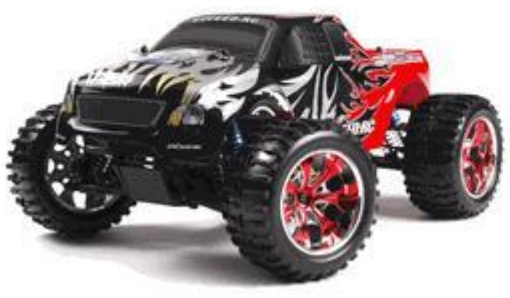


Figure 4 - Exceed-RC Infinitive EP Electric Truck

Object Detection

The remote vehicle is required to have a functioning object detection unit onboard, to ensure the vehicle can safely travel to its future location. The three best options for doing this are stereo vision, IR, and ultrasonic sensors. This section will detail what the pros and cons are for each and provide justification for the type of sensor chosen for the vehicle.

Stereo vision is one sensor type that will not be pursued due to its advanced algorithms, need for a lot of processing power, and its need for two webcams. This will add additional costs and power usage to the project, both of which are constrained resources.

IR sensors use infrared light to measure how far away an object is. They can be very accurate but this accuracy is only limited to indoors. Using these sensors outdoors will make them obsolete due to infrared interference from the sun. Due to the functional requirement for the vehicle to be fully functional outdoors, IR sensors will not be pursued.

Ultrasonic (also known as sonar) sensors use sound waves to hear how far away an object is. These can be used very well indoors and outdoors but when used indoors, can suffer from echoes from walls that will cause undue interference.

The ultrasonic sensor will be selected for use on the vehicle, due to its superior ability to determine distance, both indoors and outdoors.

To help with the aid of obstacle detection, there is already another item on the RC car: the webcam. When an ultrasonic sensor detects an obstacle, the webcam will be able to see where the object is and make a decision as how best to avoid the object. An image analyzer takes an image and determines paths on the image that avoid objects. The only concerns about using this program are how much this program will drain CPU utilization and how it will affect real-time decision making.

GPS Unit

When deciding what GPS unit to purchase, there are a number of areas you have to take in account. These are size, update rate, power, cost, number of channels, antenna, accuracy, and update rate. Below is a quick discussion of each of these areas.

- Power
  - The current average is around 30mA at 3.3V
  - Most of the power is used for analyzing data received from the satellites
- Cost
  - Vary greatly
- Number of Channels
  - Each channel represents a satellite there are a total of 24 GPS satellites and you will only see 12 at most at a time.
  - Therefore, anything above 12 Channels will be ample
- Antenna
  - Most common antennas are ceramic
  - Use power to amplify the GPS signal
  - Need to be pointed up
  - Need to be outdoors to use but can also work indoors. However, indoor operation is not

guaranteed

- Accuracy
  - Usually accurate to +/- 10m
  - Dependent on module, time of day, clarity of reception, etc
- Update rate
  - Ranges between 1Hz to 20Hz
  - Higher the update rate, leads to better accuracy when traveling at high speeds

Options

Below is a table that compares several GPS modules.

	Venus GPS with SMA Connector	Copernicus DIP Module	EM-406A SiRF III Receiver w/ Antenna
# of Channels	51	12	20
Max Update Rate	10 Hz	1 Hz	??
Accuracy	< 2.5m	< 3.0m	+/- 5m
Supply Voltage	2.7-3.3 V	2.7-3.3 V	4.5-6.5 V
Price	$49.95	$74.95	$59.95

Table 2 – Comparison of GPS Modules

Figure 5 - Venus GPS with SMA Connector

Conclusion

By looking at the table, the Venus GPS module has the best update rate, accuracy, and price of all three modules compared. Since the Venus GPS module does not come with an antenna, a ceramic antenna with a SMA connection will be purchased.

Directional Unit

A magneto sensor, also known as a digital compass, is needed for this project so the RC car's direction can be determined from a standstill. The areas of a magneto sensor that need to be analyzed are update rate and accuracy.

Options

Below is a table that compares several magneto modules.

	HMC6352	HMC5883L	MAG3110
Update Rate	1-20 Hz	160 Hz max	80 Hz max
Accuracy	0.5 degrees	1-2 degrees	??
Supply Voltage	2.7-5.2 V	2.16-3.6 V	1.95-3.6 V
Price	$34.95	$14.95	$14.95

Table 3 – Comparison of Magneto Sensors

Figure 6 – HMC6352

Conclusion

Even though both the HMC5883L and MAG3110 modules are much cheaper and offer better update rates, the HMC6352 is more accurate, has internal calibration routines, and it does not require any processing of its data from a host processor. Therefore, the HMC6352 sensor will be purchased.

Imaging Unit

To meet the requirement of having streaming video and image recognition available to the command center, a camera will have to be purchased. The camera should be supported by Linux since that is the operating system that will be running on the Panda Board. The camera should also be supported by the Panda Board, and be able to meet the minimum video quality of 240p video at 15 fps minimum with 16 bit color. In the list of supported peripherals on the Panda Board's official website only one camera is listed as being supported. This camera meets our requirements of 240p and 15 fps. So for a guaranteed implementation process this is the camera that will be chosen for the project. The camera that will be used is the Logitech Webcam Pro 9000.

Figure 7 - Logitech Webcam Pro 9000

Specifications for the Logitech Webcam Pro 9000 are the following:

- Carl Zeiss® optics with autofocus
- Native 2-MP HD sensor
- High-definition video (up to 1600 X 1200*)
- 720p widescreen mode with recommended system
- Up to 8-megapixel photos (enhanced from native 2 MP sensor)
- Microphone with Logitech® RightSound™ technology
- Up to 30-frames-per-second video
- Hi-Speed USB 2.0 certified
- Logitech® webcam software (including Logitech® Video Effects™: fun filters, avatars, video masks, and face accessories)
- Logitech Vid™
- Universal clip fits notebooks, LCD or CRT monitors

RC Car Hardware Implementation

Below is a diagram showing how all the components of the RC car and those to be added to the RC car will be connected. The types of connections are also stated in the diagram.

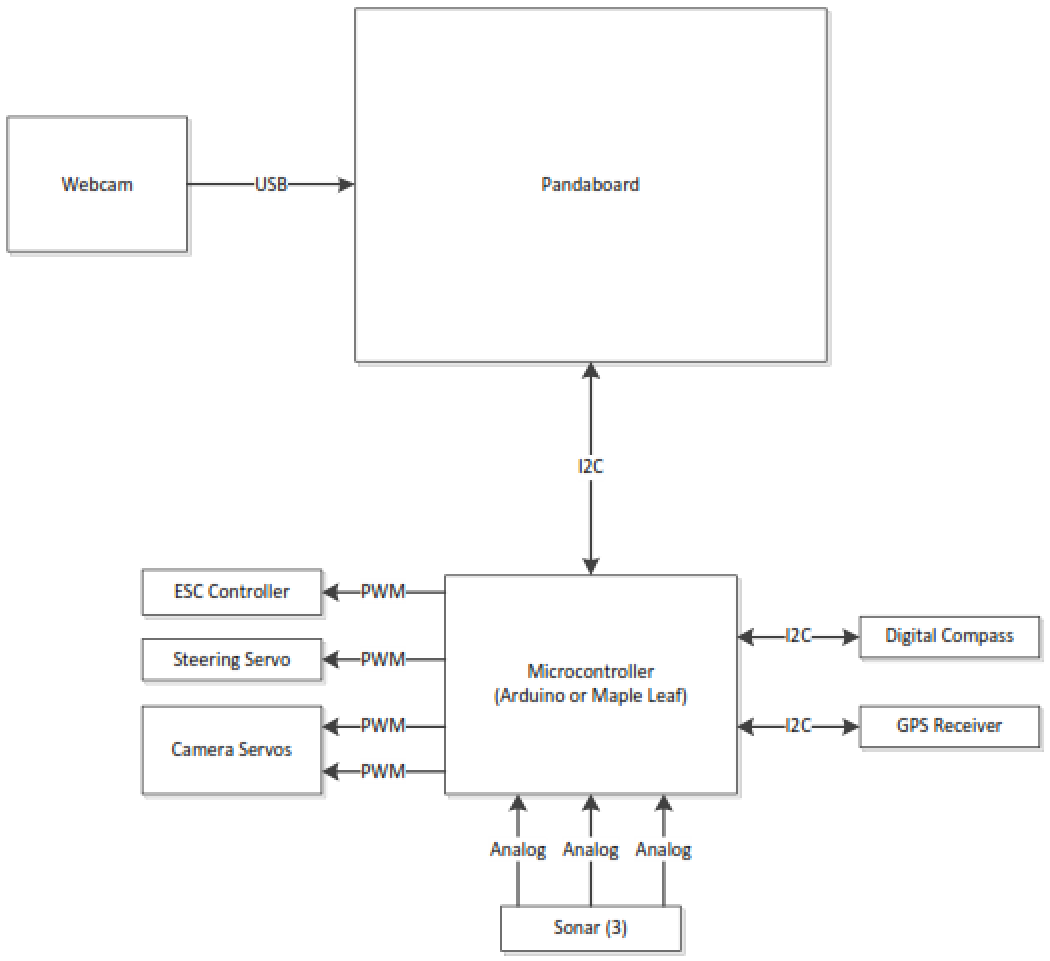


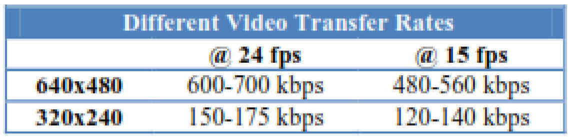
Figure 8 – System Diagram of RC Car Hardware

Communication

Zigbee

Zigbee is a new level of communication based on an IEEE 802 standard. It is best suited for applications needing to transfer data at low rates (maximum of 250kbps) and for low power consumption. Low power consumption would be ideal but the lack of data transfer will prevent the RC car to transmit any sort of streaming video. Take into account that the maximum transfer rate of 250 kbps is an optimal transfer rate. The further data has to transfer, the lower the data transfer rate of Zigbee will be.

To help get a benchmark of how much bandwidth is needed for streaming video, look at the data provided at the *Streaming Learning Center.*[3] Derived from the information at this website a summary of some different video resolutions is provided below. These are compared at 24 fps and 15 fps and their corresponding bandwidth usage is also listed. According to research done by *Streaming Learning Center*, a reduction from 24 fps to 15 fps only reduces data rate by 20% instead of nearly 50% as many would think. This shows that running at a lower frame rate would not leave a linear relationship of available bandwidth through the connection. This will be considered when working on throttling the video stream.

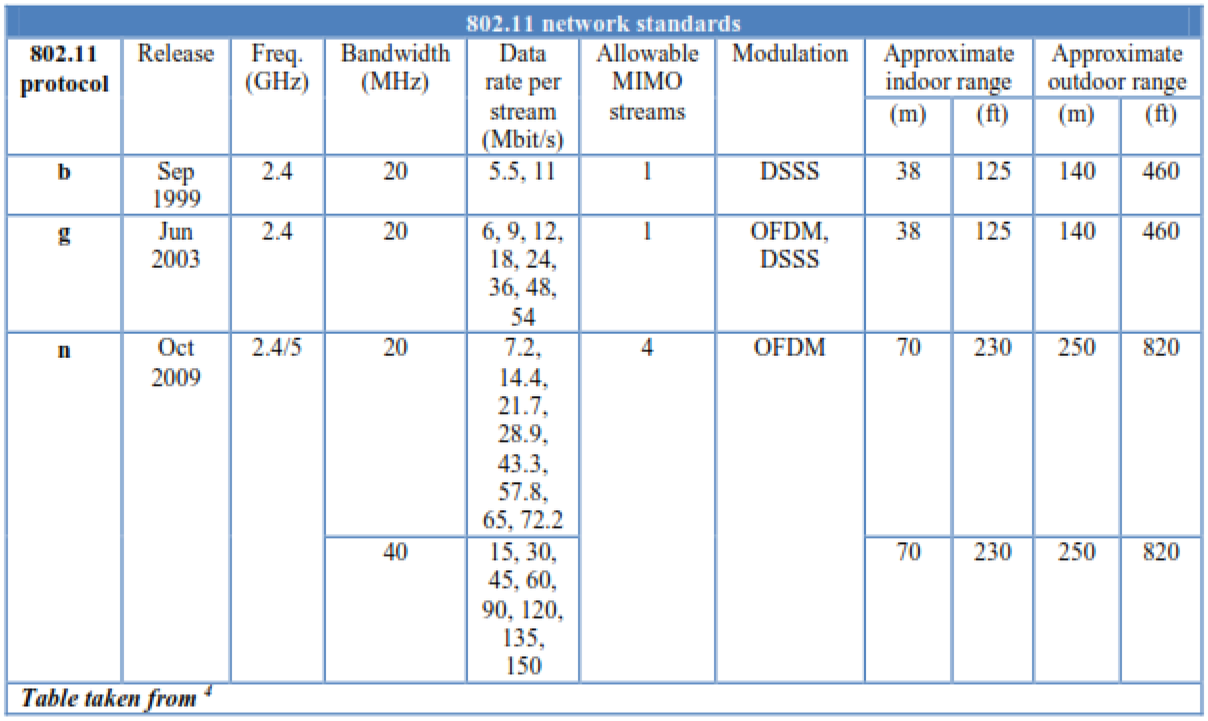


Different Video Transfer Rates		
	@ 24 fps	@ 15 fps
640x480	600-700 kbps	480-560 kbps
320x240	150-175 kbps	120-140 kbps

Table 4 – Different Video Transfer Rates

WIFI

WIFI is another option that has been looked into for the wireless communication. WIFI has a much larger bandwidth available than the Zigbee devices have and therefore is a much more viable option for the project. Also the IP protocol is natively supported in both Linux and in android, which will be the two operating systems that are used in the project. This factor would make implementation time much quicker and more reliable than creating a new software and firmware stack for the project.



802.11 network standards										
802.11 protocol	Release	Freq. (GHz)	Bandwidth (MHz)	Data rate per stream (Mbit/s)	Allowable MIMO streams	Modulation	Approximate indoor range		Approximate outdoor range	
							(m)	(ft)	(m)	(ft)
b	Sep 1999	2.4	20	5.5, 11	1	DSSS	38	125	140	460
g	Jun 2003	2.4	20	6, 9, 12, 18, 24, 36, 48, 54	1	OFDM, DSSS	38	125	140	460
n	Oct 2009	2.4/5	20	7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65, 72.2	4	OFDM	70	230	250	820
			40	15, 30, 45, 60, 90, 120, 135, 150			70	230	250	820
Table taken from [4]										

Table 5 – 802.11 Network Standards

802.11b would not be used since it is an outdated protocol and since 802.11g and 802.11n offer better data rates. In a perfect environment there would not be any interference in order to allow the theoretical data rate to become a reality. The 802.11g would provide ample range to meet the Project Plan's functional requirement of 70m range. However, since there is no way to know in advance what interference may be encountered, 802.11n would provide the greatest opportunity to meet the 70m operating range requirement.

Using WIFI the ability to route multiple cars' data through the network would increase the required bandwidth for each car, because each car would possibly rout other cars' data. This factor also demonstrates that 802.11n would be a more reliable option. The selected choice for the project will be WIFI using the 802.11n specifications and running on the 2.4GHz frequency band.

Frequency Band

Since the project will be using 802.11n, there are two different frequency bands that can be chosen from. Either the 2.4GHz or the 5 GHz band would be available. However, the 2.4 GHz band has much less interference and is implemented in all Android devices with WIFI. Thus, to ensure compatibility and cut down on interference, the 2.4 GHz band would be the better option.

Data Interchange Format

There were two data interchange formats considered for this project: XML and JSON. Both are standards and supported by all major platforms and programming languages. For this project, JSON was chosen over XML due to its smaller structure size. This means that it will take up less bandwidth to transfer across the network than XML, and will easily transfer all the information needed.

A sample JSON string for our project could look like this:

```
{ “car” : { “gps” : {
“latitude” : 42.028362 , “longitude” : -93.650959
} ,
“manual” : { “forward” : 0.47 , “right” : 0.02
} ,
“camera” : { “pan” : 0.00 , “pan_time” : 0.5 , “tilt” : 34.4 , “tilt_time” : 0.5
} ,
“sonar” : { “sonar_find” : 0
}
}
```

Transmission and Response Rates

The cars and the command center need to be in constant communication with each other. To maintain "real-time" updates between the cars and the command center, we hope to achieve 10ms latency on average between phone and vehicle. This will require a send rate of 100 times per second. The car does not need such an instant response back to the command center, and will respond at a rate of 5 times per second.

Conclusion

WIFI is the optimal solution for the wireless communication because it will provide the best bandwidth, be supported on all WIFI enabled netbooks, and be more readily available. This is the transport medium that will be used for the project.

Video Streaming

The cars will be streaming video back to the command center so that the user may view the video feed of a specific car. The video feed will be transferred over the AODVUU connection on the car. This video feed will need to be able to be scaled before it is streamed to conserve bandwidth. The scaling software will need to be supported by the Linux operating system.

GStreamer

Option one is to use Gstreamer to read in the local webcam (/dev/video0) and stream it out over the WIFI connection of the car. Gstreamer can also specify output frame rate and resolution, which will allow the feed to be scaled before the video is transmitted. The compression settings can be set up by different scripts. This feature will allow different settings to optimize the bandwidth as the available amount changes with the distance between nodes.

Custom Streamer

For option two a custom streamer application could be developed by the team. This would require a lot of time and is out of the scope of this project. Since this option would be so time consuming and require a lot more knowledge about resizing video frames, this option will be very unlikely.

Conclusion

Gstreamer appears to be the best option for the Linux operation system with the most support. Since writing a custom application to do this is so time consuming, the project will use Gstreamer.

Command Center Design

Platform

When choosing a platform to act as the command center node, the main considerations are networking capabilities and screen size. Network capabilities include which wireless protocols the device supports, and how the client implements them. Screen size is important due to the functional requirement to stream video and still images from the vehicles to the command center device.

IEEE 802.11n is an industry wireless standard that is an evolution of 802.11g, which is an evolution of 802.11b and 802.11a. All evolutions are backward compatible. Every standard uses the 2.4 GHz frequency spectrum, except for 802.11n, which also supports the 5 GHz spectrum. Chips that support both the 2.4 GHz and 5 GHz spectrum are labeled as being “dual-band”.

Android

Some Android manufactures have their devices listed as only supporting 802.11b/g, while others only list 802.11n networking chips in their models. Due to software limitations, no Android phone is natively able to support connectivity to an ad-hoc network.

A development called Wi-Fi Direct is a promising contender for native ad-hoc support and is implemented into the next iteration of Android, called Ice Cream Sandwich. However, this is untested and too unknown for consideration in this project.

- Pros
  - Some phones may support ad-hoc networking
  - Mobile
  - Easy to develop for
  - Larger selection of devices
  - 4 hardware buttons
- Cons
  - Most/all phones will not support ad-hoc networking without rooting

iPhone

The iPhone 5 has hardware support for 802.11b/g/n. Like the Android operating system, the iPhone’s operating system does not allow for iPhone to iPhone connections or an ad-hoc network to be established. In both Android and iPhone, Bluetooth can be used to create a direct connection between devices; however, it only has a max range of 100m and a throughput of 24 Mbps, which is not acceptable for the project.

- Pros
  - Mobile
  - Low variability between versions
- Cons
  - Will not support ad-hoc networking
  - Mac OS is needed for development
  - $99 fee for developer account
  - No options for devices except for amount of memory
  - Only one hardware button

Netbook

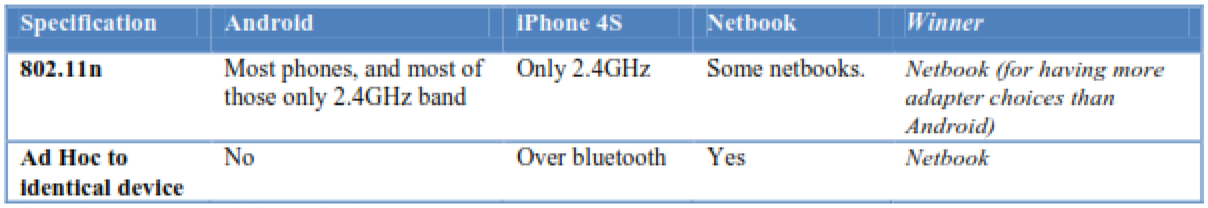
Most recent netbooks support the 802.11b/g/n protocols. Whether the netbook supports dual-band 802.11n communication depends solely on the wireless NIC card inside the netbook, but there are many on the market that support it. Furthermore, there are no software limitations on the netbook, which means that ad-hoc networks can be created between many netbooks.

- Pros
  - Supports ad-hoc networking
  - Better WIFI capabilities (range, frequency)
  - Wider range of programming languages
- Cons
  - Not as mobile
  - Higher power consumption

Comparison

When choosing what device to use as our command center, we chose the best one that fit into our previous specifications and allowed for the most options going forward. WiFi is the wireless communications standard for our project; more specifically, IEEE 802.11n radios will be used for their superior bandwidth and range. All the choices have options available with 802.11n adapters in them.

Since one of the project's highest priorities is to connect to a mesh ad-hoc network, we researched the ad-hoc capabilities of each device. Of the three, only the netbook is able to create an ad-hoc connection to an identical device. An Android phone cannot connect over 802.11n to another Android phone directly, nor can iPhone to iPhone. For this reason, the netbook is a better solution for integration into an ad-hoc network.



Specification	Android	iPhone 4S	Netbook	*Winner*
802.11n	Most phones, and most of those only 2.4GHz band	Only 2.4GHz	Some netbooks.	*Netbook (for having more adapter choices than Android)*
Ad Hoc to identical device	No	Over bluetooth	Yes	*Netbook*

Table 6 – Comparison of Mobile Platforms

Conclusion

We will be using a netbook for the command center. Having better WIFI capabilities and ad-hoc support is the number one priority. Developing in Java on a netbook will enable development in many programming languages. This gives easy portability on to PC, Mac, or Linux because Java is run on a virtual machine compatible across platforms. Java also offers easy multi- threading and networking interface. Another reason for selecting a netbook is the keyboard and mouse. These extra inputs options will allow the UI to free up valuable screen space.

Netbook Selection

The netbook selected for this project is an Acer Aspire One. The netbook runs Ubuntu Linux and will give us the greatest flexibility. Developing in Java on a Linux netbook will also allow us to more easily port the command center over to an android tablet to meet our lowest priority.

Architecture

The software in this project will have to tackle several issues. One of the biggest will be communication over the network. We will have to send and receive messages from other cars and the command center. The cars will have to send sensor updates and streaming video on demand and the command center will have to receive this data.

Communication

The command center will use a monitoring architecture that will listen to a socket for messages. The mission commands will be sent using TCP and the video will be streamed using RTSP. We chose TCP for messages because they have guaranteed delivery. It is critical that the missions are delivered. All of our sending and receiving of mission and sensor data will be done through a communication module. All of the communication methods will be housed here. This will eliminate the risk of communication changes affecting the rest of the source code.

Any sensor that is added to the car will implement the Sensor interface. This will allow for expanded functionality in the future. Any sensor will be able to be added and accessed from the device. Having this requirement allows for a low cost in adding more features.

Command Center Interface

The command center is comprised of two major screens. The first is the Mission screen that has a Map interface that shows the route of all robots, sensor values for each robot, and a button that brings up a screen to select mission priority levels. The second is a screen that shows only information for a specific car. This information includes current sensor values, Map of current location, and a button to turn on the video feed. At the top of the screen there is always a button for each car and a button to return to the mission screen available on every screen.

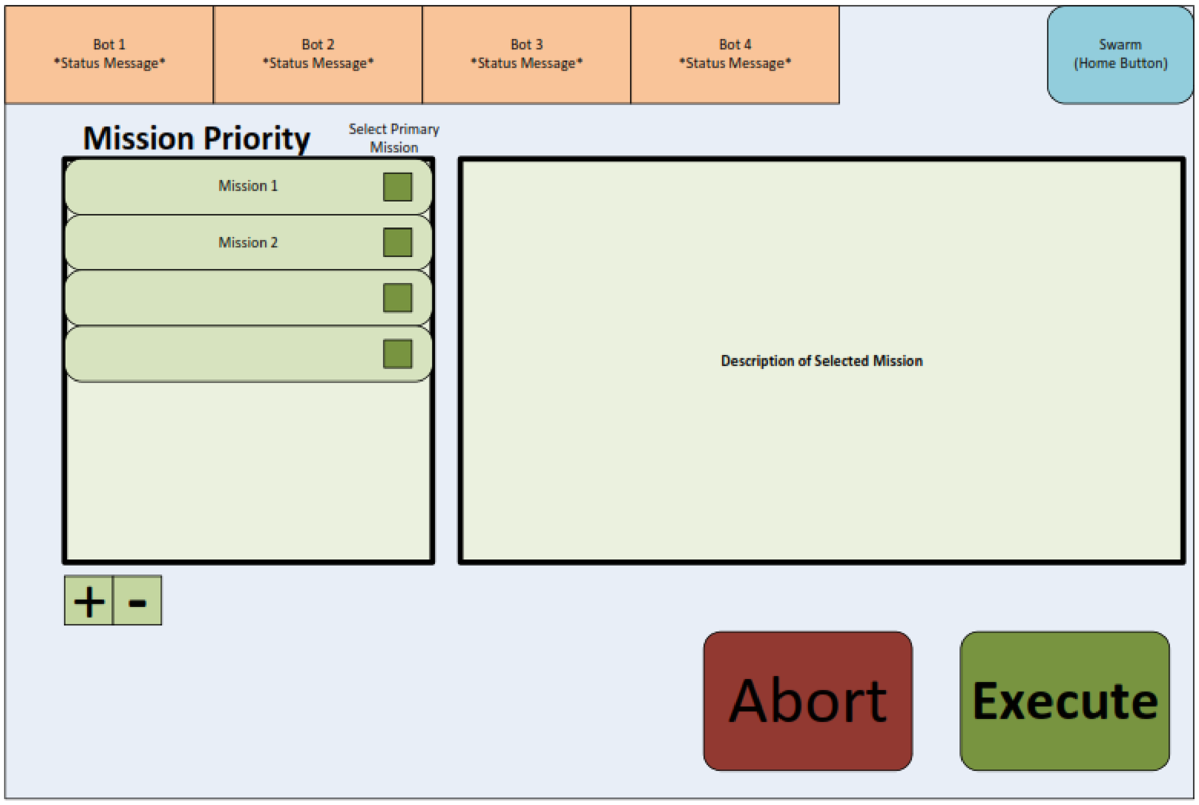


Figure 9 – Mission Priority Screen

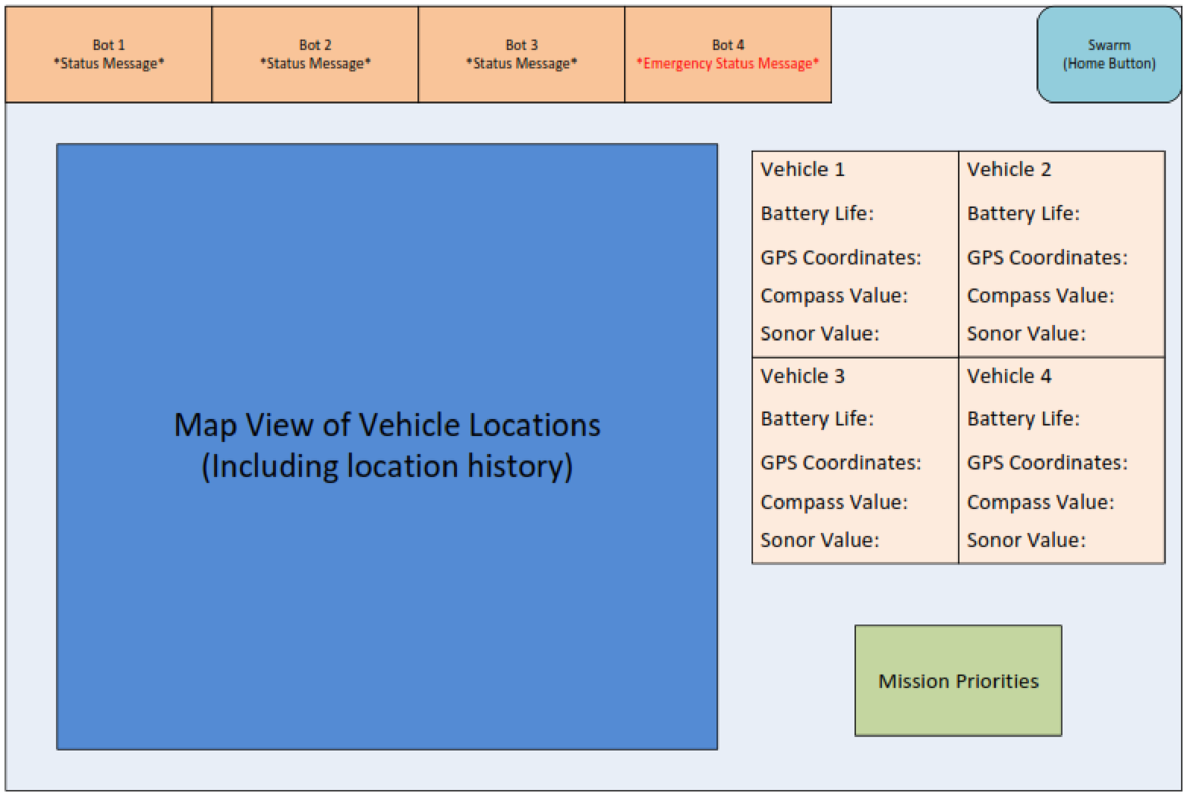


Figure 10 – Swarm Screen

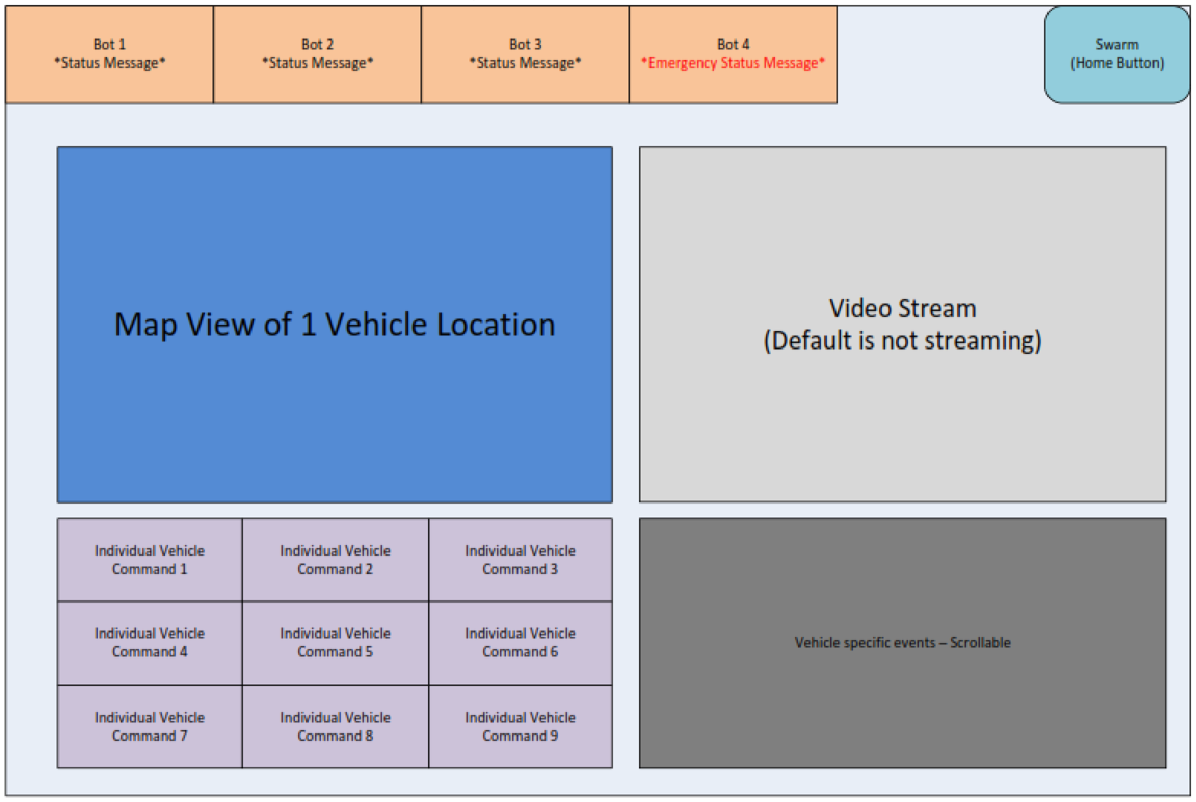

Bot 1
Status Message
Bot 2
Status Message
Bot 3
Status Message
Bot 4
Emergency Status Message
Swarm
(Home Button)
Map View of 1 Vehicle Location
Video Stream
(Default is not streaming)
Individual Vehicle Command 1
Individual Vehicle Command 2
Individual Vehicle Command 3
Individual Vehicle Command 4
Individual Vehicle Command 5
Individual Vehicle Command 6
Individual Vehicle Command 7
Individual Vehicle Command 8
Individual Vehicle Command 9
Vehicle specific events – Scrollable

Figure 11 – Vehicle Screen

Reference Documents

[1]http://tirokartblog.wordpress.com/2011/03/26/beagleboard-xm-v-PandaBoard-running-syntrowebcam/

[2]http://en.wikipedia.org/wiki/Brushless_DC_electric_motor

[3]http://www.streaminglearningcenter.com/articles/streaming-101-the-basics---codecs-bandwidth-data- rate-and-resolution.html

[4]http://en.wikipedia.org/wiki/IEEE_802.11

http://en.wikipedia.org/wiki/Bluetooth#Uses

http://www.iphonedevsdk.com/forum/iphone-sdk-development/7282-ad-hoc-wifi-connection-between- two-iphones.html

http://www.apple.com/iphone/specs.html

http://www.howardforums.com/showthread.php/1716949-Android-Phones-with-5Ghz-%28Dual- Band%29-Wifi-chips

http://en.wikipedia.org/wiki/Comparison_of_Android_devices