Embedded Systems Development

Project Artefact – Particle Photon Swarm

Name - Jack Murley

Student ID - 218294029

Table of Contents

1. Overview

Within Swarming the cost of each individual member/node in that swarm is roughly 150$ and when the average swarm now is considered of up to 100 000 nodes the overall cost of that project is at minimum 150 000 00$ USD for a large scale swarm. Thus as a result of this price tag on swarms it greatly holds back the development of both low scale and large scale swarms primarily due to the cost associated with testing your algorithms on real robots.

Currently the industry standard is Harvard's Kilobot. Harvards Kilobot is roughly 150$ USD per each robot. Below is an image of the Kilobot:

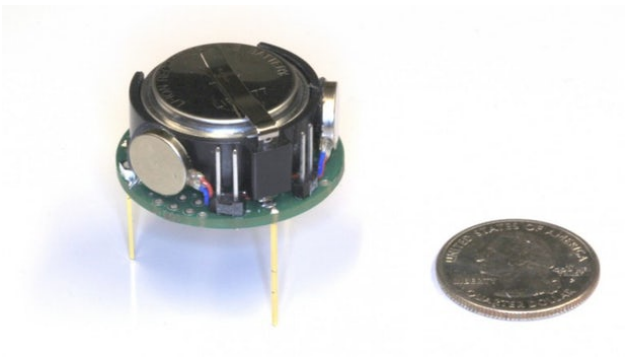

Figure 1: Harvards Kilobot

The aim of my project is to not only create a embedded system that is capable of doing basic swarm theory however also create a device that is significantly cheaper than Harvard's as well as has larger capability's through the ability to constantly interchange sensors on each node in the swarm.

As for the purpose of testing the proof of concept swarm nodes that I have created they will be required to do the following actions to be considered as a real swarm and not an IOT (Internet of Things) swarm. These tasks are the following:

- Detect each other member of the node as well as know where they currently are in relative context to a specific node
- Move as one and stay in formation while moving together
- Be able to change formation based upon the objects in the way of the swarm
- Map out the current environment that the swarm is moving in
- To achieve the set goal that the swarm has been given and successfully achieve that goal within a reasonable time relative to the complexity of the goal

The swarm devices I have created uses a particle photon connected to 4 infrared object detection sensors to detect objects as well as other nodes in the swarm. It is currently a two dc motor-wheeled rover with a support wheel at the front, which is controlled by the L298N Dual Motor Driver, which is powered by a 4 double AA battery pack, which also powers the particle photon. The swarm moves as one and stops as one and changes formation based upon the surroundings as well as changes based of off the other members of the swarm.

2. Background/Problem Statement

Swarming is currently a massive field within robotics that is ever changing however numerous issues and factors hold it back. One such factor that plays a major role in holding back developing advanced swarming technology is cost as mentioned above. While researchers might get grants of up to 5 million dollars from various sources such as the government, the defence force, industry and individual investors it does not leave a lot of room for individuals to develop embedded systems that are able/capable of performing large swarming features as they simply do not have the money to send.

While cost is one factor another major factor is system design. Currently swarming devices are high advanced and unable to interchange sensors such as Infrared sensors as well as ultrasonic. Thus as a result if you want to test different algorithms that require other sensors you are required to completely design an entire new device, which is built specifically towards that algorithm. Ultimately this creates an endless cycle where you have to completely work on a single algorithm until you achieve it or risk having to lose the progress you have made towards that algorithm, this can be seen in both the academic as well individual sector when making swarming technology.

However within swarming, especially large-scale swarms a major issue ironically is space. Most swarming algorithms currently require a specific dedicated space to run and test the swarms movements and functions. While many researchers can easily find the clear perfectly flat space and simulate almost a perfect environment many individuals working on/experimenting with swarming simply cannot spare the extra space. This particular evident within todays housing market as many individuals now live with family and thus as a result don't have a spare room or dedicated area to directly assign to just swarming.

Thus as a result my embedded system aims to combat 2 of the 3 of the main issues identified above. This is primarily because space is particularly hard to overcome within large-scale swarms, as you require a small enough device like the kilobit that is still as durable as the honeybee swarm robot.

My embedded system aims to combat the issue of cost associated within large scales swarms as currently my swarm at the current state is only 30$ per node, where as Harvard's (seen as the industry standard) is currently 150$ each. Ultimately which results in a 500% decrease in price for the individuals using my embedded system as the base of the swarm versus Harvard's Kilobot.

The embedded system I have designed also aims to combat system design as the system I have developed has the ability for a variety of sensors to be interchangeable with the Infrared sensors currently on the system. Some possible replaces for the Infrared sensors are the following:

- Ultrasonic
- Ambient Light Sensors
- LED's

Thus as a result this reduces the need of creating multiple devices for different algorithms as my embedded system allows for the interchangeably of sensors which in turn reduces the cost associated with large scale swarms for multiple algorithms.

3. Requirements

For a multi-robot system to be considered a swarm and not an IOT swarm they have to meet the following criteria:

- Be classified as a Heterogeneous or homogenous swarm of robots
- Be able to self-reconfigure its formation/range
- Be able to communicate to each node in the swarm either directly or indirectly through other nodes
- Communicate to each other in a timely manner
- Scalability, the ability for the swarm to change size by either increase or decrease based upon how nodes are active
- Robust, Be able to continue operation regardless of the failure of a single node
- Be able to react and think based off of multiple nodes and not communicate through master slave communication

The following criteria are rather important as it sets out and lays down a guideline to ensure that the embedded system that I create is truly a swarm and not a multi-robot system.

As well as the criteria above the following operation criteria will also have to be met during testing to ensure that the swarm has a high enough level of functionality to be a working swarm. The criteria is as follows:

- Be able to communicate to each node of the swarm within less than 0.5 seconds each way
- Be able to move as one without comprising each other nodes safety
- Be able to change formation based upon the readings from the environment
- Be able to track its movement and map its environment while on the fly and as a result provide a effective 2d map of objects in the current environment
- Be able to protect the leader of the swarm and prioritize the safety of the leader over other nodes in the swarm

For the swarm to operate certain requirements also have to be met, these requirements are the following:

- Flat surface this is due to the wheel base of the swarm as well as the power of the DC motors
- Active and strong internet connection this is due to the SOC (particle photon) as it requires an internet connection to run any code
- A 15 volt power supply, this is to ensure that the motors as well as the sensors have enough power to provide the sensors with accurate data
-

4. Design Principles

While designing my embedded system various design principles and ideas were significantly considered during the development. However the most important factor behind my design of my embedded system is to have the most weight towards the back near the dc motors.

This is primarily because if the weight was primarily at the front not only does it risk a high likelihood of the system being to heavy on the support wheel and not moving due to the added friction however due to the wire length.

Thus as a result is why the embedded system is built on 3 different layers. These layers are:

- Top Layer, where the sensors are positioned
- Middle Layer, where the SOC and Motor Controller are positioned
- Bottom Layer, where the DC motors and Battery Pack are held

The Layers are specifically put together so the top layer has the least amount of weight and the bottom layer has the most amount of weight associated to it. This is primarily because the top layer is put together with a single piece of flexi glass and 3 pillar supports, which go into the bottom layer.

Thus as a result it is essential that the largest amount of weight is on the bottom layer to decrease the possibility of the support pillars breaking or even cracking/warping the flexi-glass. This is key to ensure, because if the flexi-glass warps then the sensors on top of the system also get a warped view. Ultimately in turn could result in the sensors not getting an accurate reading.

Furthermore the design allows easy and quick access to the key components. As the top layer is easily removed which in turn allows an individual to access the SOC and motor controller to make reasonable changes. This key to ensure that the design has a high level of modifiability as well as flexibility in what swarming algorithms can be run on the device.

The decision to house the battery pack on the bottom of embedded system is to ensure that it leaves and provides enough space for extra SOC or even another motor controller on the middle layer to ensure that the embedded system can be converted to a 4x4 if needed.

Nevertheless, the pillar supports, which hold up the top layer, are also strategically positioned to ensure safety for both the motor controller as well as SOC. This is done to ensure that if the motor controller losses or for some unforseen reason comes off of its pillars it is still able to be held in a confined space which will minimise the risk of a wire possible coming out of the motor controller.

Likewise the same has been done for the SOC to ensure that it will decrease the risk of the embedded system stopping or malfunction while in the field or possibly testing.

Below are the following schematics, which show the interconnections between the sensors and the SOC as well as the motor controller and DC motors.

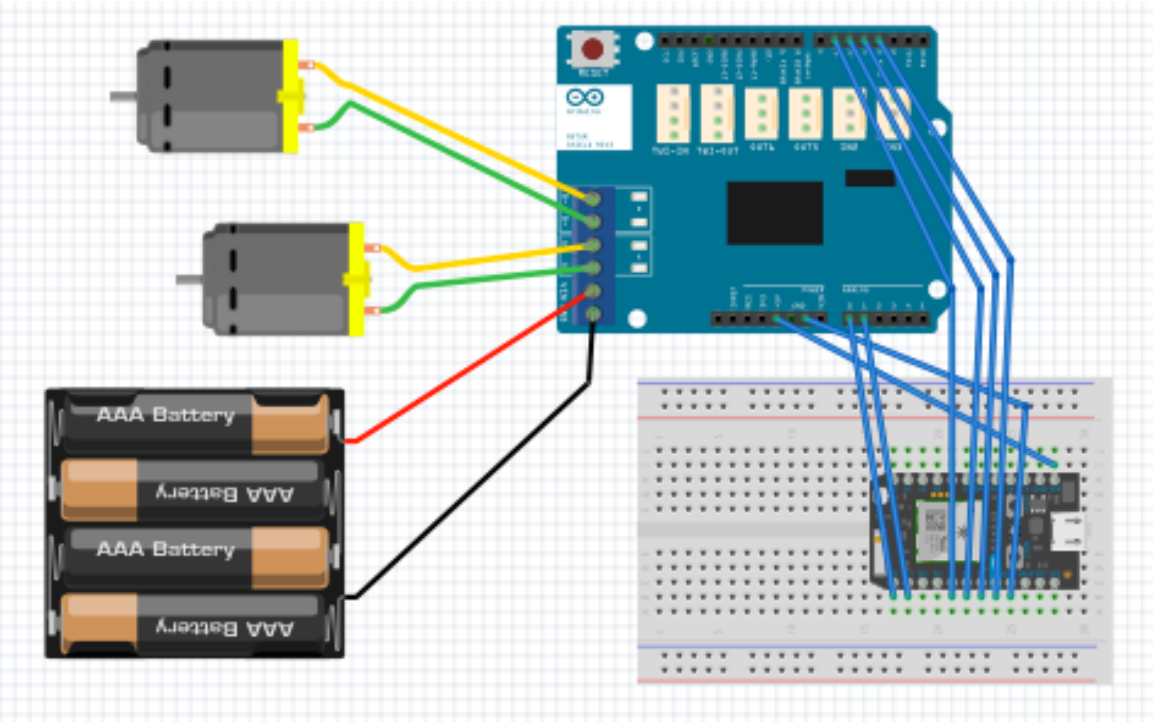

Figure 2: Schematic of the motor controller wiring

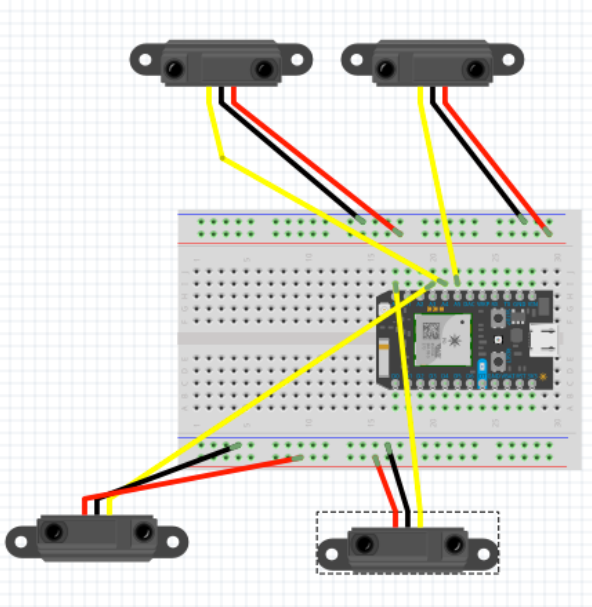

Figure 3: Wiring of IR sensors

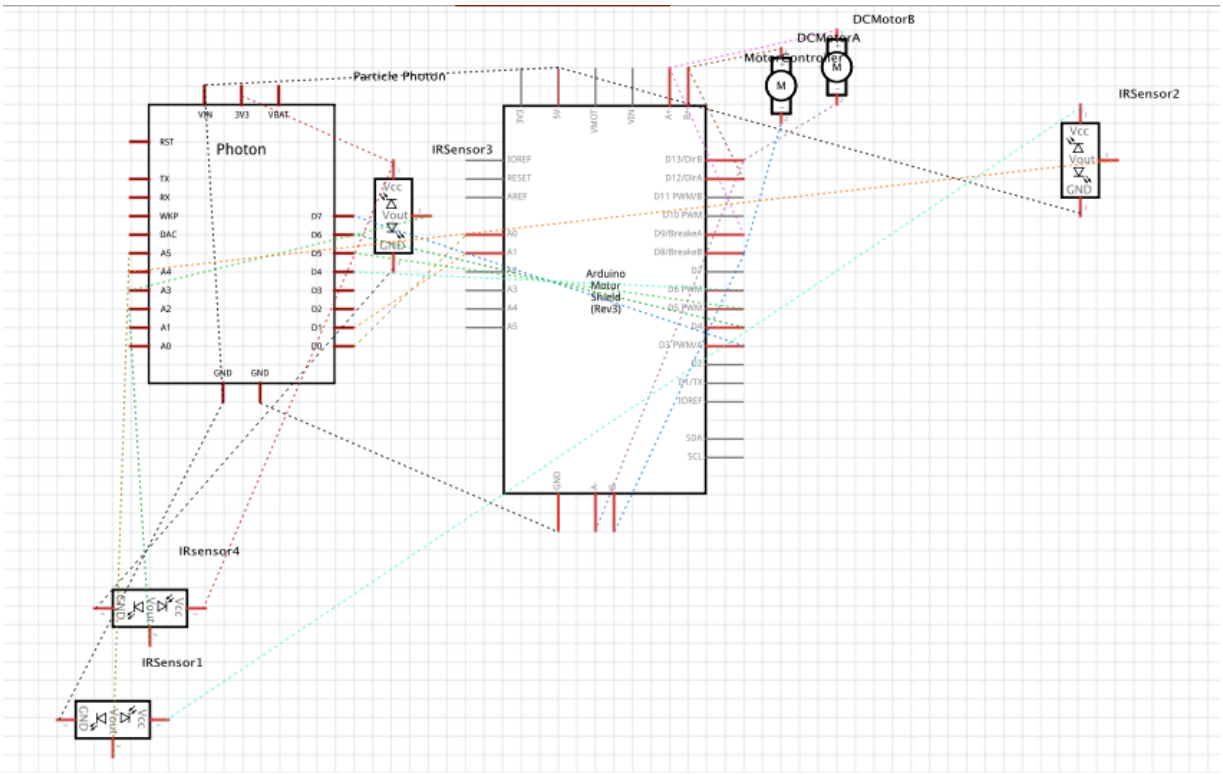

Figure 4: Sample overall schematic

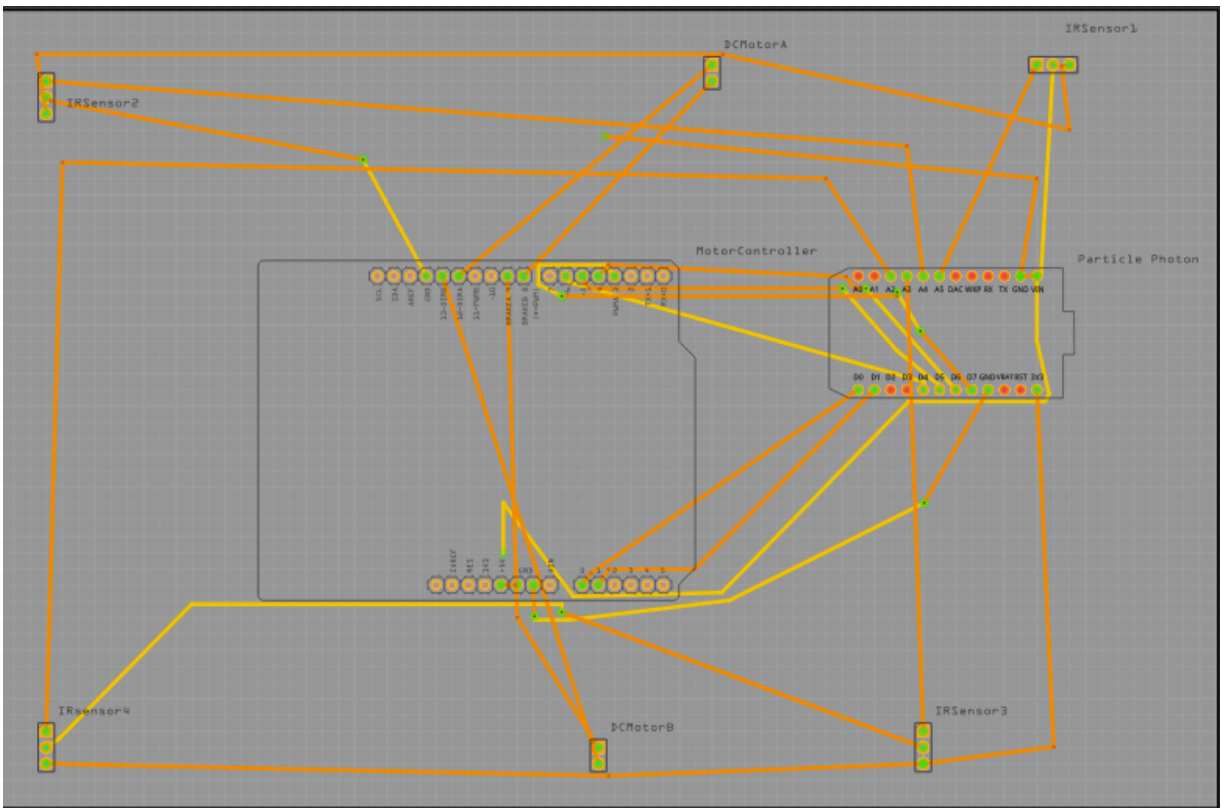

Figure 5: Sample PCB design

5. Prototype Architecture

My embedded system currently uses the IP/TCP protocol to communicate to each of the nodes. However the specific SOC that I have implemented uses an IP/UDP (User Datagram Protocol) to send and receive data as this is an alternative to the TCP and is used to establish low latency and loss-tolerating connections between each nodes. This is particularly useful in a swarm environment as the faster you are able to send and receive data to all nodes in the swarm the higher likelihood you have of no node being left in the dark essentially.

Furthermore the embedded system uses Google's geo-location api to track where each node in the swarm currently is and plots onto a virtual map their exact location when they detect a object which is not another member of the swarm. Which in turn creates a simple SLAM algorithm of the environment that they move around in

6. Prototype Code (GitHub)

The overall embedded system source files can be found online from my personal GitHub repository, which is found at the following URL –

In total we have 3 code files, 1 for the leader of the swarm, 2 for the other nodes of the swarm. This is particularly important as they do significantly different things thus as a result require different code to efficiently function.

Leader Code:

https://github.com/JTMurley/SIT223/tree/master/LeaderCode

Left Node Code:

https://github.com/JTMurley/SIT223/tree/master/LeftNode

Right Node Code:

https://github.com/JTMurley/SIT223/tree/master/RightNode

If you would like for the swarm to act different for example say instead of moving the motors turn on an LED all you would simply have to do is go within the specific loop functions and some of the clean up code and change it to turn on an LED instead.

7. Testing And Evaluation

To test my embedded system certain requirements had to be met for my embedded system to be considered a swarm. Thus as a result the first test that was conducted on my embedded system was to see if the nodes of the swarm could communicate to the leader and if the leader could communicate to all the nodes as this is at the heart of swarms the most important requirement to have.

To test this requirements the motors of each node were turned off and each node was put in range of each other's to test and see if the geo-location as well as the sensors sent the correct data to each other member.

Once it was establish that the geo-location as well as the sensor data sent to each member of the node allowed the swarm to communicate to each other the testing of the geo-location failsafe was then tested.

To test this particular feature we left two nodes of the swarm in place and then took one away roughly 60 CM to see if the geo-location picked up that a member was taken away while someone was trying to either emulate a member of the swarm still being their or if a object was in place which emulated a member of the swarm.

Ultimately from the results of this test it showed that not only did the geo-location feature work but it also ensured that the swarm could not be emulated easily by someone or by an object.

The final test that was conducted on the swarm was to see if they could move as one as well as change formation based upon the surroundings. The results from this test showed a significant design fault in the embedded system. This design fault being that the support wheel caused a member of node to go off course and not go directly straight or backwards in a straight enough to maintain communication between the nodes.

However if you hold each member of the swarm you counter react this effect and as a result they move like expected. Therefore while the test for the movement of the swarm was a failure the proof of concept proved that the underlying code was a success. Therefore in a perfect environment/simulation the swarm will work like expected. However when testing in a closed environment the swarm will fail due to the design flaws.

8. User Manual

In order to run my swarm and the features that are capable within the underlying code you will need the following environment:

- High speed networking to ensure the publish and subscribe events can be sent in a timely manner that does not negatively affect the swarm
- A almost perfectly flat surface
- A open environment
- A operating environment which produces minimal friction
- A fully charged battery pack in all members of the swarm

Once the following criteria has been met you will need to ensure that the leader of the swarm is placed in the middle of the two other nodes, within this scenario we are assuming that the swarm only consists of 3 members, the leader, the left and the right side.

After the swarm has been placed appropriately in the correct environment you will need to ensure that you turn on the two slaves in the swarm first and then proceed to turn on the leader once the slaves have established a Wi-Fi connection.

Once all members in the swarm are turned on and in place they will proceed to move as one in a straight line, however if they are not held to counteract the support wheel they will quickly lose formation and stop moving as one and as a result stop moving as a whole because they have detected that they are not in formation.

9. Conclusion

Ultimately my embedded system not only allows for a cheaper large-scale swarm but also enables more flexibility when wanting to design for multiple swarming algorithms. However while saying this, the embedded system I have designed is not entirely fault proof and this can be seen through the movement of my swarm. Due to the support wheel at the front of each node of the swarm and how it swerves it ultimately causes the nodes to not maintain a straight line while in action.

Thus as a result causes the swarm to lose formation as well as struggle to change formation efficiently as well as effectively due to this design fault. However due to the nature of my embedded system design the support wheel can easily be interchangeable for any type of support wheel or even a new wheel base such as a 4 x 4 or the use of treads.

Therefore we can conclude that the overall aim of my embedded system succeeded in achieving the set out goals. These goals were:

- To create a open source embedded system cheaper than the industry standard
- To create an open source embedded system that allows for the interchangeability of various sensors without compromising the overall design.
- To create a embedded system which individuals could easily implement and build themselves

However as we say this, significant improvement can still be made upon the embedded system. Some improvements could be the following:

- Add a camera to one of the nodes of the camera to allow for a third reference point
- Implement a drone above the swarm which provides real time data back to the members of the swarm from a birds eye view
- Improve overall stability of the swarm
- Improve the swarming algorithms and create them scalable which allows for any size swarm