A HARDWARE & SOFTWARE OPTIMISED SWARM PLATFORM

ABSTRACT

In current swarm systems, quadcopter are commonly used as flight platforms. However they are unable to be effective platforms due to the lack of optimization present and this project aims to solve that problem by proposing a hardware and software optimized swarm platform. By designing a unique frame which encloses the electronics and increases durability and resistance to mid-air collisions, we managed to make a hardware optimized swarm platform. By implement a mesh network and pooled computing, we managed to also optimize the software of the platform. Results indicate that the optimization greatly helps to improve performance by allowing the swarm to be more dense due to the more compact and resilient design. We are also able to widen the applications as the pooled computing allows the platform to make complex decisions otherwise not possible with individual computers. This could drive the swarm intelligence industry to greater heights as an optimized swarm platform could be critical for research applications

CONTENTS PAGE

INTRODUCTION

This project aims to achieve a hardware and software optimized swarm flight platform that allows for swarm developers to have an easier time developing and using swarm platforms in real life applications.

Current swarm platforms mostly use conventional quadrotors for their applications. However, conventional quadrotors have several problems when confronted with swarm activity. These problems inhibit the abilities of these swarm platforms as their structure inherently prevents the close interaction of swarmbots in swarm behavior.

Quadrotors use four propellers to generate thrust and variable speed control for movement. The inherent design has it that there are 4 propellers spinning at high speed which are blatantly exposed. In a swarm environment where the drones have to fly in close proximity, these propellers are highly likely to cause a mind air collision.

Our swarm platform, SentiBot, uses a new drone design to solve the inherent problems with the quadrotor while maintaining the versatility of the quad. It also adds resilience and durability to the frame. Close-range flying is no problem with this frame as all the moving components are contained within the frame.

The SentiBot platform also uses intelligent software which optimizes the performance by allowing the SentiBot swarm to pool information and computing power.

By creating an optimized platform for swarm research, we hope to accelerate the progress in swarm research and allow for swarm research to be applied in industrial and civil applications.

HYPOTHESIS

"A hardware and software optimized swarm platform will increase the performance and abilities of current swarm algorithms by providing a more capable platform for research"

SENTIBOT SPECIFICATIONS

Dimensions:
100mm x 100mm x 250mm

Flight time: 10 - 15min

Motor thrust: 450g

Payload weight: 100g

Top speed: 20km/h

HARDWARE OPTIMISATION

We have implemented several new designs and strategies to optimize the hardware performance on the SentiBot. By using the novel frame design, we are able to contain the delicate electronics inside the drone while also providing a certain element of modularity. By implementing a custom electronics design, we are able to shrink the control and processing electronics to make the design more compact. These improvements coupled with the software give us a highly optimized swarm platform.

FRAME DESIGN

The novel frame design on our SentiBot platform is one of the reasons for its ideal characteristics for swarm. Our frame design goes against the conventional design of 4 motors by replacing it with a single high power **E**lectronic **D**ucted **F**an (EDF). This reduces the high-speed moving parts on the design and therefore lowers the complexity and chance of failure. By using an EDF instead of a propeller design, we also managed to get much better static thrust performance which is ideal of hovering in these types on drones. For control, we forewent the differential thrust and went for 4 control surfaces. These control surfaces work much rather in the same way as airplane flaps and direct air to achieve control. By enclosing the whole design in a bean shaped frame, we enable the frame to contain all the essential electronics inside the frame and prevent damage to the moving components.

By using the bean-shaped frame as a reference point, we were able to further improve the design by adding modularity. In a research platform, modularity is very important as users of the platform may want to add other sensors to enhance the robot's functionality. Therefore, we split the design in the top half and the bottom half. The top half of the frame is dedicated to the "power" electronics. This includes the EDF, **E**lectronic **S**peed **C**ontrol (ESC) and the battery. This half can be modified in the case of motor failure of other power problems and allows for easy trouble shooting in case anything goes wrong. It also allows for upgrades to the power train and thus allows the user to increase or decrease the payload and speed abilities of the drone as required.

The second half of the frame will be used to contain the "intelligence" electronics. These include the control electronics, processor and the servos. This section can be modified to add sensors and modify the code in the ATMEGA 328 control chip. The 2 halves are connected by a joining piece which you bolt into a captive nut. This allows for easy modifications to the frame without intense modifications to be made to the frame.

Lastly, the frame design allows easy for large-scale manufacturing, The 3D files which make up the components in the frame are simple 3D shapes that can be easily molded and cast thus making it easier to manufacture for research purposes. As of now the frame uses a polylactic acid (PLA) material, but this material could be changed to more durable ones like polycarbonate or Nylon-6.

In conclusion, the frame design allows for a durable and modular swarm platform which is able to be easily implemented into swarm systems.

HARDWARE OPTIMISATION

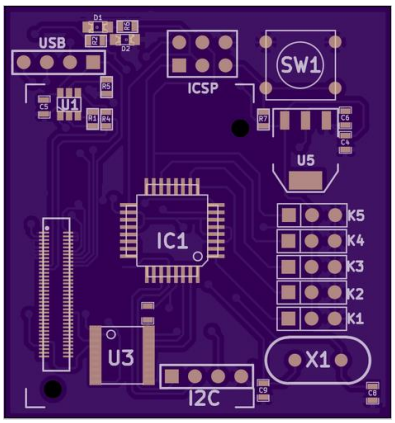
ELECTRONICS DESIGN

The electronics design is a critical part of the platform as the electronics provide the foundation for the control system of the robot. Since the robot is intended more use in computing power intensive applications like search and rescue and swarm intelligence, it is paramount that we have a processor on board. Our processor of choice was the Intel Edison. It provides a decent amount of computing power operating at 500MHz with 512MB of RAM. It also has 4GB of onboard storage for any applications or programs. Since it runs a version of Debian Linux, Ubilinux, it makes it very easy for you to program it.

However, the Intel Edison comes with a very small SMD connector which we were going to need to breakout. Furthermore, we realized that we needed a separate microcontroller to manage the control logic to avoid over pressure on the Edison. We ended up using the ATMEGA 328 microcontroller chip and flashing the control logic onto it. We created a custom circuit board for breaking out the Edison connector and to route in the ATMEGA 328 chip. We also provided output for the USB camera and the servo for controlling the flaps.

The control loop also required an Inertial Measurement Unit (IMU) to detect orientation and compensate for oscillations present in the frame. By using an I2C interface to the ATMEGA 328, we managed to connect the ArduIMU v3 to the chip which allow for the chip to compensate for oscillations. This will also help to stabilize the frame as seen in the following section. The ArduIMU v3 is connected to the PCB through a set of header pins which allow the IMU to either be temporarily or permanently connected to the PCB.

The power electronics consists of 3 1C 1300mAh batteries, a 20A ESC and an EDF. The EDF has a max current draw of 18A and taking into consideration that the max thrust of the EDF is 450g the gram per current rating of the EDF is 25g/A. Since the weight of the SentiBot is about 200g, the nominal current draw during a hover would be about 8A. That gives us a flight time of about 10-15min. This is a pretty respectable amount of flight time for a conventional drone. This configuration can of course be changed in case the user required more power or more flight time.Lastly, I would like to talk about the power management system in the SentiBot. We will be using the in-built BEC in the ESC which provides clean 5V out to power the board. The board also has a onboard 3V regulator to provide power to the Edison and the logic level converter which allows UART communication between the 1.8V Edison and the ATMEGA328 for the Edison to send high level commands to the ATMEGA 328 controller.



This ecosystem provides a hierarchy system where a high level Edison processor is able to send commands for the low level controller to provide autonomous control of the drone.

SOFTWARE OPTIMISATION

Software optimization is an important part of this project as we can use the software to improve the characteristics of the drone in swarm behavior and use the nature of the SentiBot to our advantage. In the following sections we can see the algorithms and communication techniques used to facilitate the operation of the SentiBot.

ALGORITHMS

Control Algorithm

The control algorithm used in the SentiBot is similar to all the other control algorithms used in typical quadcopters. Implementing the classic PID control loop, we allow the sensor data to be mapped into error values that can be used to compensate for frame errors and instability in the frame. The control algorithm first reads the I2C bus and outputs the data from the IMU which has been preprocessed by the Digital Signal Processor (DSP) on board the IMU. This allows for the control algorithm to get filtered and processed yaw-pitch-roll (ypr) data for use in the PID algorithm. The processed ypr values are sent through the respective PID loops for their specific axis and used to obtain 3 error values for each of the x, y and z axis. These error values can then be used to output to the servos, which then tries to bring the system back to the stable state by moving the specific servos. All of the control algorithm runs onboard the ATMEGA 328.

Vision Algorithm

The vision algorithm is the use of the single camera in vision to obtain navigational data for navigating and positioning in unknown indoor environments. Since we only have space onboard for a single camera, it is not possible to carry out stereoscopic mapping of the surroundings. Therefore, we resulted to using a monocular simultaneous locationing and mapping (SLAM) algorithm for navigating around in unfamiliar environments. This was done by installing ROS into the Edison flashed with ubilinux. We then used one of the many available monocular SLAM algorithms available integrating it such that it uses the roserial library to send high level serial commands to the ATMEGA 328. This algorithm is still being tested on as the time of writing this report.

Swarm Algorithm

The swarm algorithm as not been implemented into the SentiBot yet but it has been conceptualized and is ready for testing. The first algorithm we are going to test is one for maze solving. The mesh network allows it to communicate information between the bots that allow for it to create a more accurate and bigger map of the area it is exploring. This algorithm would treat each group of bots as a swarm. So, when it has 2 or more paths to explore, the swarm would split equally if possible. Each swarm is assigned a leader. When a swarm splits, the leader of the swarm will assign each of the new swarms a new leader. If the number of bots is lesser than the number of paths, the location of an unexplored path is marked. When a swarm meets a dead end, it would communicate to the other swarms. If there are any paths left unexplored, the leader communicate to the previous leader. If the leader has any unexplored paths, the swarm would follow the path back to the point of splitting and then follow the path to the explored path it is assigned to by the previous leader while also marking the path as exploring. Otherwise, the swarm would converge with the other swarm with the path provided by the previous leader. Upon finding the exit, all the bots would be recalled unless specified to find multiple exits.

SOFTWARE OPTIMISATION

COMMUNICATION

The communication system of the SentiBot is a unique design for optimizing the performance of the swarm system. There are primarily 2 types of communication between SentiBots which is computer pooling and a mesh network. I will be detailing both methods below.

Computer Pooling

The computer pooling model optimizes the swarm performance by allowing the SentiBot swarm as a whole to act as a supercomputer. This is done by making each SentiBot a node in a system of nodes which forms the basis of a supercomputer which is connected via Wi-Fi. This pooled computing power therefore allows the swarm to make complex decision as a whole. Instead of taking the conventional method with swarms where each robot is an individual being which interacts solely with self-interest, this method allows the swarm to think as a whole where the sensor data can be pooled to allow the swarm to survive as a whole instead of prioritizing the survival of each individual,

Furthermore, the pooling of computer power allows for the processing of the vision data to be done off the individual processor and instead be done in the supercomputer “cloud”. This may improve performance of the vision system however this can to be verified with further testing. This supercomputer system can also be used in other applications like search & rescue algorithms which is computing intensive. Other possibilities include a neural network running in this supercomputer to recognize objects and other vision applications.

Mesh network

The mesh network model also makes the SentiBot a node in a system of nodes however it establishes a peer to peer connection between them which enables a mesh network of SentiBots. This mesh network frame work can be used as a range extender for the user to receive streamed data from the autonomous swarm. This can be achieved where some SentiBots can act as “active” participants carrying out the mission while other SentiBots can act as “passive” participants who simple relay signals through themselves thus allowing the SentiBot to “toss” information around to one another until it ends up with the recipient.

This can extend beyond the user as the swarm can use it to decrease the density of the swarm when required. For example, in situations like search and rescue where the search area is large, this allows the swarm to spread itself around the search area without losing interconnectivity easily.

RESULTS

The results of initial testing were quite positive. The SentiBot is able to remain in constant flight even without any PID stabilization algorithms and remain in relatively oscillation free flight when there is PID stabilization. Furthermore, the drop tests were significantly positive showing that the SentiBot can survive drops from up to 3m high due to its flexible and rigid Nylon-6 frame.

In testing, the top speed achieved from the SentiBot is also about 20km/h and flight times are about 10min with normal flying. Furthermore, the SentiBot is able to achieve versatile flight through, under and over several obstacles without any problems due to its size and form factor. The SentiBot can also easily take collisions with walls or other obstacles without falling.

The vision results are still inconclusive as the algorithms are still being tuned at the time of the writing of this paper. However, as of now, the SentiBot is able to carry out some amount of 3D mapping though we are fine-tuning the algorithm to get a greater amount of regulation.

The swarm and communication algorithms have not been tested in real life yet but have been somewhat simulated and tested out virtually. Simulations seem positive and work will be done on it after the completion of this report.

CONCLUSION

The SentiBot project set out to bring a solution to swarm researchers worldwide. We aimed to provide for all the hardware and software requirements for a swarm system. The results seem to indicate that we have made considerable progress towards that goal but there is further testing and tweaking to be carried out in the following years.

The results also seem to indicate that the hardware side of swarm research is just as important as the software side as the hardware optimization solved many problems with current flying swarms. We hope that this report can act as a kickstarter for hardware development in swarm research.

Future progress could be made in this field. Some interesting applications that this could be implemented into is search and rescue, reconnaissance and research platforms. We recommend a future study into further optimizing the swarm platform for specific fields and also doing further research on the software and communication optimization techniques outlined above.

In conclusion, this was a relatively successful project that shows how both hardware and software needs to be brought together to achieve he perfect swarm system and also highlights the lack of research into hardware optimization for swarm platform.