SEMI-AUTONOMOUS RESCUE TEAM

Team Description Materials

Riley Cockerill, Ryan Ewyk, Aaron Maggs, Jack Williams & Matthew Williams
contact@sfxrescue.com

Logistical Information

Team Name	Semi-Autonomous Rescue Team	
Organisation	St Francis Xavier College, Florey, ACT	
Country	Australia	
Mentor	Gerard Elias	
Contact Person 1	Graham Stock	
	Phone:	+61 417 439 452
	Email:	Graham.Stock@sfx.act.edu.au
Contact Person 2	Matthew Pham	
	Phone:	+61 420 760 341
	Email:	Matthew.Pham@sfx.act.edu.au
Website	https://www.sfxrescue.com	

1.0 Abstract

Figure 1 - The Semi-Autonomous Rescue Team, 2015 - 2017

The Semi-Autonomous Rescue Team (herein known as the S.A.R.T.) (Figure 1) is a small group of Robotics and Information Technology enthusiasts formed in late 2015 with the intent of creating a Robot capable of competing in the 2016 Rapidly Manufactured Rescue League (RMRL) at RoboCup in Leipzig, Germany.

The project started with the basic Robot design and specifications supplied by Curtin University in Western Australia. The idea was to use this as a starting point onto which teams could innovate and rework into their own Robot. The original S.A.R.T. Robot consisted of the basic chassis design from Curtin University known as the "Emu Mini 2", paired with a Raspberry Pi B+ and Dynamixel AX-12A servo motors. Over the past eighteen months, the initial design has transitioned to a fully custom designed and 3D printed chassis, an Intel NUC to replace the Raspberry Pi as the main control board, an Arduino Nano for collecting sensor data and an oCam 5 megapixel USB camera for streaming the Robot's point of view to a Control Panel Interface over enterprise grade Wi-Fi. With these improvements, the team has experienced great success in competition, including:

- Tied 1st Place – Rapidly Manufactured Rescue League, RoboCup 2016, Leipzig, Germany
- 1st Place – Rapidly Manufactured Rescue League, RoboCup 2017, Nagoya, Japan
- Open Source Award – Rapidly Manufactured Rescue League, RoboCup 2017, Nagoya, Japan

2.0 Introduction

After forming in late 2015, the Semi-Autonomous Rescue Team entered its first competitive event in 2016 at RoboCup in Leipzig, Germany. The Robot's original design was based on the specifications and design supplied by Curtin University in Western Australia for the Emu Mini 2. This design was improved before the competition to include streaming camera footage from a Raspberry Pi Camera to a computer and controlling the Robot itself over a short-range Bluetooth connection using a PlayStation 3 controller. The camera stream enabled the remote operation of the Robot when the operator does not have a direct line of sight, or, in the case of RoboCup, when the operator cannot look at the Robot while competing.

One of the major issues with the Robot while competing in Germany was the latency of the video stream due to the massive congestion in the wireless radio environment – the delay made it difficult to navigate the Robot through the courses accurately. The congested wireless environment inspired a greatly improved Wi-Fi network utilising an industrial grade Access Point (AP). Implementation of such a bulky device would require an enclosure, as well as a

computer to control it, so a Control Panel with a mini PC, AP, Uninterruptable Power Supply (UPS) for the Power over Ethernet (PoE) and related peripherals was constructed. The UPS also allows the Control Panel to be portable so the Robot can be operated in remote conditions or without access to stable power. Observing the performance of other Robots at the competition inspired changes to the Robot itself. One such improvement was to modify the chassis for upgraded internal components arranged in such a way that the space inside the chassis was used more efficiently by the more powerful hardware.

After observations of the Robot's performance and the performance of other teams in the RoboCup 2017 competition, it was concluded that a Robot performs better with a longer wheelbase and larger wheels, up to a point. Having an excessively long wheelbase limits the Robot's mobility in the courses, and the maximum size of the wheels directly correlates with the length of the wheelbase. Other improvements to the software side of the Robot's operation include real-time motion tracking and image recognition as well as more accurate temperature and distance measurement, as the current temperature sensor can only reliably indicate a dramatic change in temperature rather than the quantitative value. The Infrared (IR) distance sensors cannot seem to be configured to return an accurate measurement that makes sense. As for the Robot's movement, the fine control over direction and speed afforded by the Control Panel was helpful in navigating through the courses. The improved controls allow the operator to drive the Robot efficiently through a course or turn it around by flipping over against a wall to travel in the opposite direction without having to spin on its axis in a confined space.

3.0 System Description

3.1 Hardware

3.1.1 Robot

3.1.1.1 Chassis Design

The physical design for the Robot went through nineteen different iterations of the chassis and wheels in order to make it suitable to be rapidly manufactured on commercially available 3D printers.

The basis of our original design objective of creating a Robot chassis for this competition was to make something modular, so that different components could be swapped out if broken or if the use case called for a more specialised part. Everything was to be self-contained and capable of being put together like Lego, with modules attaching with a universal 'snap in' mechanism. After a few prototypes were designed and 3D printed, it was concluded that while the Robot was modular, the physical dimensions were too large to fit in the maze. Additionally, the singular point of contact for the snap in connector was not robust enough to withstand what we would consider normal use. The additional parts to 3D print drove up the manufacturing time and cost. Overall we determined that, as a proof of concept, modularity on this scale was impractical for real-world applications.

Starting from scratch, we looked at the original design of the Open Academic Robot Kit's Emu Mini 2. Considering its pros and cons, we realised that the form factor was ideal for navigating through the competition maze and the wheelbase allowed it to turn on the spot. With all the new hardware we intended to use in the Robot, a total redesign was required to ensure everything could fit in a similar footprint to the original Emu Mini 2. Simulated test fits were run on the nineteen different prototypes of the current chassis model, which involved using 3D models of the components (such as the Intel NUC, SSD, Camera, Arduino and battery) to verify that everything would fit inside the smallest package possible.

While the basic rectangular design was inspired by the Emu Mini 2, some of the original features were omitted in favour of other features that we considered more important. One of the more notable modifications was the exclusion of a mechanical arm so that the Robot was symmetrical on both top and bottom. The rationale behind this decision was that the Robot could drive up against a wall and flip over. In our testing, we discovered that this was quicker than turning around, especially when the dimensions of the competition maze are too small for the Robot to spin on its axis. The other notable change from the original Emu Mini 2 was the versatility of the servo mounts. Servos on the Emu Mini 2 could only be mounted in a single orientation due to the screw

holes and power/data cable pass-through holes (Figure 2).

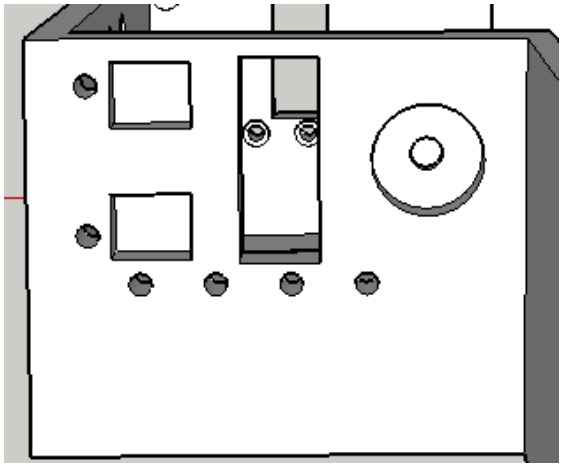


Figure 2 - The original servo mounts (from the Emu Mini 2) only allowed the servos to mount in a single orientation.

The Emu Mini 2 design differs to the design that we created, which had a single, larger cable pass-through hole and more screw holes where the servos mounted, giving us the flexibility to orient the servos however we wanted (Figure 3).

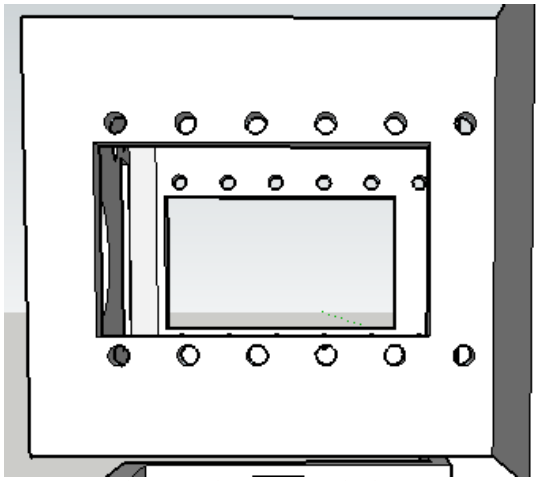


Figure 3 - The new design for a dual orientation servo mount. The only shortcoming of this design is the reduced strength of the chassis as a result of the rectangular holes in the sides where the greatest twisting forces are experienced.

Our design gave us three options for wheelbase length potentially having a short, medium or long wheelbase by simply changing the orientation of individual servos.

Unlike the Emu Mini 2, we did not add any extra holes or unnecessary embossed text as they provided an entry point for foreign objects, increased print time and made it more likely to fail.

The final chassis consisted of a rectangular box of dimensions 221.10mm long, 150mm wide and 50mm high (Figure 4).

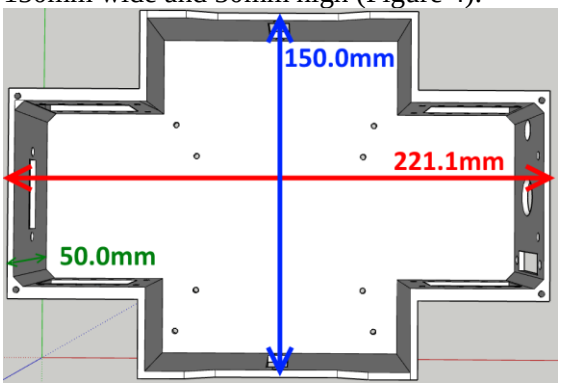


Figure 4 - (Top View) The dimensions of the final Robot chassis.

It had cutouts for attaching the servos at two orientations, and a lid that prevented the internal components from falling out or foreign objects from entering the chassis. The physical size was the smallest that we could make it, given that we had to fit more powerful and larger hardware into it – mainly the small-form-factor computer, SSD, power delivery, battery, camera and the multitude of sensors, compass, accelerometer, gyroscope and Arduino.

3.1.1.2 Internal Components (Figure 5)

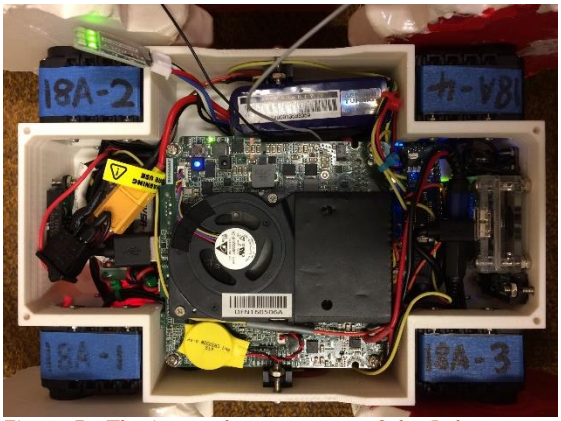


Figure 5. The internal components of the Robot.

The main computational component of the Robot (i.e. The brain) is the Intel NUC5CPYH which features a 1.6GHz dual core Intel Celeron N3050 processor. The more powerful brain (compared to the Raspberry Pi we used in 2015/2016) allows for high frame rate, high bitrate and high resolution streaming, which also takes advantage of the superior and interchangeable Wi-Fi antenna on the NUC compared to the fixed antenna on the Raspberry Pi.

The sensors (Infrared distance, gyroscope, compass, accelerometer) connected to an Arduino Nano which, like the camera was

connected to the Intel NUC via USB. This means swapping out different components uses a backwards compatible, universal standard connector, and, is what we used to interface the servos with the NUC via a USB2AX.

Power was supplied by a lithium polymer battery with a capacity of 1.3 Ah, giving around an hour of continuous use, powering the servos and NUC while streaming to a single client. The limited power supply was one reason that the NUC5CPYH was chosen, as it had a good power consumption to performance ratio, whereas a Pentium, i3 or i5 model would require a larger capacity battery to run for the same period of time.

The camera was the oCam 5MP USB 3.0 Camera. Although it has the capability for image processing built in, we only used it for streaming. In the future, some image processing can be offloaded to the camera rather than the NUC or Control Panel.

Under our open source philosophy, everything was made free to use under the GNU-GPL license on the 3D model sharing website *Thingiverse.* Also uploaded to *Thingiverse* was editable versions of the wheels, chassis and belts so other people can use the S.A.R.T designs as a starting point for their own Robots or as a source of inspiration or modification.

3.1.2 Control Panel

The idea behind the S.A.R.T. Control Panel is that everything needed to operate the Robot is contained within a single package (Figure 6).

Figure 6 - A conceptual render of the S.A.R.T. Control Panel, created to aid the communication of the idea before construction began.

Inside the Control Panel is an Uninterruptable Power Supply (UPS), Xirrus XR620 Access Point, BreezeLite Fanless Mini PC, AOC widescreen LED monitor and a Microsoft wireless keyboard and trackpad all contained in a Pelican carrying case (Figure 7).

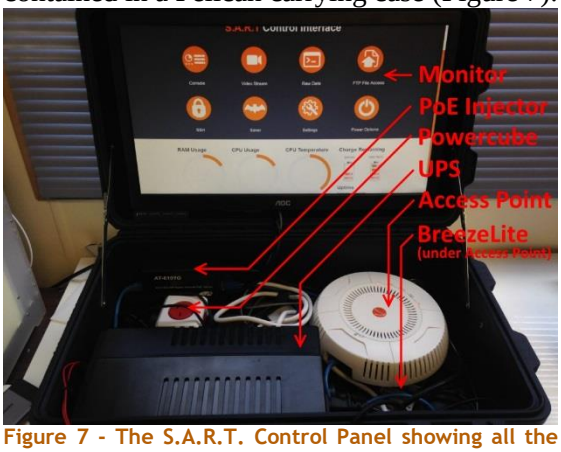


Figure 7 - The S.A.R.T. Control Panel showing all the internal components.

The BreezeLite Mini PC was chosen based on its low power consumption and variety of I/O, solid state flash storage, small form factor and its passive cooling capability. It features a quad core Intel Atom x5-Z3250 processor, part of the lowest power consuming skew of Intel's mobile processors at only 2W of typical power consumption. The solid state flash storage meant that there were no moving parts that could be damaged if the Control Panel experienced a shock, unlike hard drives where a significant shock or drop can cause the head and spinning platter to misalign and, as a result not function correctly or at all. The small form factor of the BreezeLite Mini PC meant that it could fit inside the case, leaving enough room for the Access Point, UPS and power delivery apparatus.

The Xirrus XR620 Access Point was chosen for our applications based on its customisability and reliability, providing a fast, stable connection despite a considerable amount of external wireless interference. Despite the fact that it is a higher end Access Point and being rather expensive, it was given to us by our school so we would not have any issues with Wi-Fi unreliability, range or dropped connections.

The decision of the specific model of UPS we chose for the Control Panel came down to how long the battery would last based on our workload within the size constraints of the Pelican case. In Japan, we found having a UPS immensely helpful to convert 110 volts to the 240V Australian standard that we required for everything to work properly, with the aid of a transformer.

The screen we used was a generic 1600x900 resolution display that was chosen simply because it would fit inside the lid of the case and because it had a VGA connector which we used to connect it to the BreezeLite Mini PC. The advantage of using a display over VGA rather than HDMI (also supported by the BreezeLite) was that VGA cables have screws, so the cable would not come loose from the screen or BreezeLite during the competition.
The wireless keyboard used to control the BreezeLite Mini PC (and subsequently the Robot) was a generic Microsoft device chosen mainly for its slim form factor (allowing the Control Panel lid to close without having to remove the keyboard) and the included touchpad, negating the necessity for a mouse. The operator can simply use a finger to point around the screen, whereas a mouse requires operators to potentially have to deal with uneven or poor reflective surfaces that can cause tracking problems with a traditional optical mouse.

Despite our use of a dedicated Control Panel, any device can be connected and used to operate the S.A.R.T. as long as the Robot has been previously instructed to automatically connect to the same network as the device used to control it. This is because the Control Interface is hosted on the Robot's local web server, and all the user needs to do is enter the Robot's IP address in the URL bar of their browser and have access to the control, streaming and data collection functionality.

3.2 Software

3.2.1 Robot

Ubuntu 16.04 LTS was used as the Robot's operating system. It was chosen as a lightweight and more open alternative to Windows operating systems, and as a friendlier alternative to other Linux distributions. Its x86/64 architecture made it capable of running on the desktop-grade hardware in the Intel NUC. Choosing a free operating system also allows others to replicate our project without having to pay for an operating system such as Windows.

The code we run on top of Ubuntu is coded in Python, chosen because of its versatility, ease of use and popularity. Some additional Python libraries were installed, such as Pyax12, AsyncIO, WebSockets and OpenCV.

There are 6 main Python scripts run on the S.A.R.T.

motion_tracking_filtered.py is a motion detection and tracking script. It takes frames from the video stream and analyses them for objects that change position over time. If detected, it bounds the object in a box for easy identification.

performance_party_data.py collects system performance data using the Python library psutil. The information is formatted and sent to the Control Interface via WebSockets, where it undergoes further formatting and is displayed in the system monitoring section.

qr-read.py is a QR code reader that takes frames from the video stream and displays decoded text in almost real time.

sensor_party_server.py is a simple WebSocket server that listens for sensor data from the Arduino Nano's serial port. The data is formatted and sent on to the Control Interface.

servo_party.py contains a WebSocket server that waits for instructions from the Control Interface. When a key press has been received, the instructions for the appropriate action are sent to the Dynamixel servos.

servo_party_data.py is a WebSocket server that reads temperature and supply voltage data from every connected servo. The data is formatted and sent to the Control Interface, where it is used to calculate battery life statistics and report servo temperatures to the operator.

In addition to the servo control scripts and software, the Robot runs Motion, a free software package designed for home surveillance and motion detection. It is important to make the distinction between motion detection, where the program simply detects a change in pixels, and motion tracking, where the program is intelligent enough to track something throughout the scene and interpret it as a single object. We had originally used a branch of Motion specifically for the Raspberry Pi called MotionPi, so when we migrated to the NUC in late 2016, we continued to use Motion (albeit the master branch rather than the Pi-specific version). We chose to continue with Motion because of our experience and because the Control Interface was already working with the software.

The Robot runs the popular web server *Apache 2* to serve the Web Control Interface. The interface was originally written in PHP,

CSS and HTML. With the recent adoption of WebSockets for communication, PHP has been phased out and replaced with JavaScript for the majority of communication and processing.

Like our 3D models, all our code was made free to use and edit under the GNU-GPL license on *GitHub*, where we encouraged other teams to contribute to or find inspiration in our solutions.

3.2.2 Control Panel

The Control Panel (in the context of software) can be defined as any device that connects to the S.A.R.T. Robot and can load the interface in a web browser. Such a device can run any operating system (mobile devices may work theoretically, given a keyboard is paired with the device, although this is untested). The officially supported browser is Google Chrome, which allows the user to use all features of the Robot. Mileage may vary when using other browsers.

Additional functionality may be unlocked by using the official S.A.R.T. Control Panel, as it includes local software for extra features. The official Control Panel runs Windows Server to manage a DHCP server to assign IP addresses to devices connecting via the Xirrus AP, however, any operating system will work if a DHCP server is installed. The official Control Panel also includes a default gateway system with information that aids the user in setting up the Robot. A built-in IP scanner allows the user to check what IP the Robot is connected to in the event that the DHCP server is not functioning.

No other software is necessary on the Control Panel, as the interface itself is hosted on the Robot's local web server.

3.3 Communications

The backbone of our network is the industrial grade Xirrus XR-620 Access Point, with hundreds of configuration options allowing us to edit our network as we please. The basic communication settings we used during the RoboCup 2017 competition were as follows:

Band: 5GHz
Wi-Fi Mode: A
Channel: 44
Channel bonding: Disabled
Antenna count: Up to 4.

The settings we used in the competition are by no means the limit of our access point's capability. It is capable of running 2.4GHz to 5GHz Wi-Fi bands in modes ranging from ac, a, b, g and n and delivering transfer speeds of up to 1.7Gbps to 240 individual clients. It is also capable of filtering out other Wi-Fi networks to ensure that it is the dominant network, which could be useful in a congested area. It can dynamically change channels depending on the congestion and traffic and also can bond those channels together for a more stable and reliable connection for greater range and signal strength.

The Control Panel runs a DHCP server that has allowed us to connect devices to the network at random. This also means we do not have to manually assign a device an IP address as the DHCP server does that automatically. Another advantage this gives us is to set reserved IP addresses. This is done to lock the Robot to one IP address that is always assigned to the same device. The static IP address means that scripts that communicate between the Robot and Control Panel do not have to be updated with new IP address variables.

The advantage of using an industrial grade access point is that with all the bandwidth is concentrated on a single client, meaning that it is less likely to experience an outage. Because access points of this class are designed to be reliable in congested areas, the benefit to using this access point is because it helps guarantee a more stable, reliable and stronger signal, even in a congested environment similar to the one experienced at RoboCup 2017.

3.4 Human-Robot Interface

First responders in a rescue situation rate ease of use highly on what they want from a confined space rescue Robot. They need to be able to control the Robot with minimal coding experience and prior training. This design philosophy was incorporated into the S.A.R.T. from day one, with the human-Robot interface built from the ground up with ease of use in mind.

The culmination of this effort is a web-based Control Interface that can be used on any device with a compatible web browser. It is a complete front end experience that removes the operator from the bare bones of the operating system via a sleek interface (Figure 8).

Figure 8 - The main page of the S.A.R.T. Control Interface

The WASD keys, the most popular left-hand key mapping for arrow keys, are used to control the Robot's basic movement. The number row (1 through to 0 on the keyboard) is used to adjust the Robot's speed in increments of 10%, affording fine control to balance torque, speed and momentum during a mission. A keystroke collection service runs in the background, meaning the operator can use these controls while viewing any number of draggable and rearrangeable windows (Figure 9). This ability to choose what is displayed and where allows the operator to see only the information they want on the screen, with no distractions.

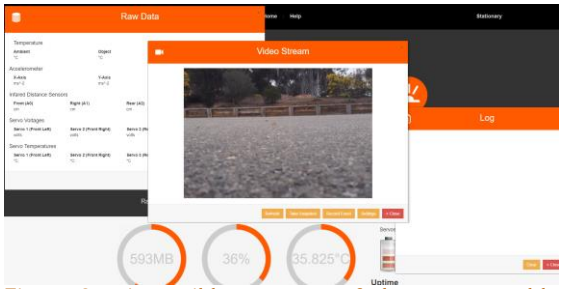


Figure 9 - A possible use case of the rearrangeable windows. In this case, the operator chose to view the raw data, video stream and log at the same time.

The Control Interface features 10 main distinctive features.

The **SSH Terminal** (Figure 10), powered by *ShellInABox*, gives the operator complete access to the S.A.R.T's Ubuntu operating system. This can be useful for many things from killing unresponsive processes to monitoring activity on htop.

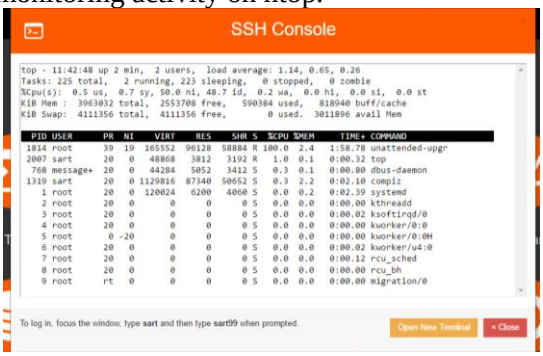


Figure 10 - The SSH console running the process monitor "top".

The **Video Stream** (Figure 11) is powered by a highly customised software package called Motion. Our modifications disable the motion detection (not motion tracking) features for faster processing, and modify the quality to size ratios for optimal streaming performance. The result is a latency free, high-quality video of everything the Robot can see, streamed directly to the operator in real time.

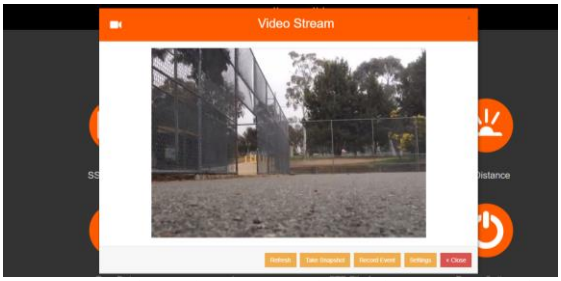


Figure 11 - The video stream shows the operator what the Robot can see in the middle of a mission.

Audio Communication (Figure 12) is a relatively new feature, only implemented during RoboCup 2017. A Text to Speech system allows the operator to talk to a victim nearby the Robot, while the Speech to Text system allows the victim to talk to the operator. In the future, this system will be further streamlined using the VoIP protocol to allow bi-directional communication similar to a phone or Skype call.

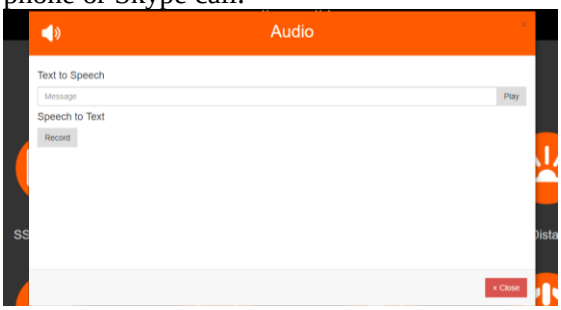


Figure 12 - The audio feature allows the operator to communicate with the victim and vice versa.

IR Distance (Figure 13) is a planned mapping feature that makes use of the 4 infrared distance sensors on each side of the Robot. It shows the operator the distance between the Robot and the nearest obstacle.

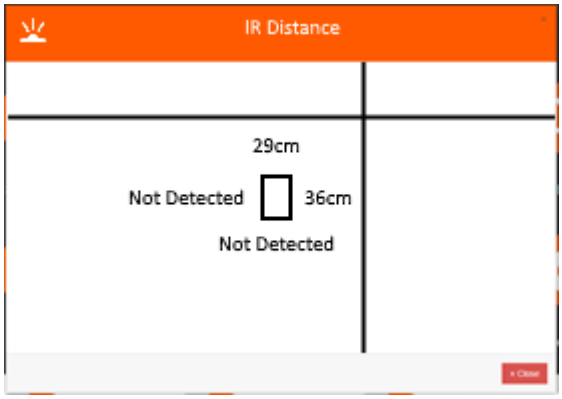


Figure 13. The IR sensor display indicates the distance to the nearest obstacle on each side of the Robot.

The **Raw Data Output** (Figure 14) window displays various sensor and servo statistics in text form. This can be used for anything from checking if an individual servo is overheating to reading the temperature of an object in front of the Robot.

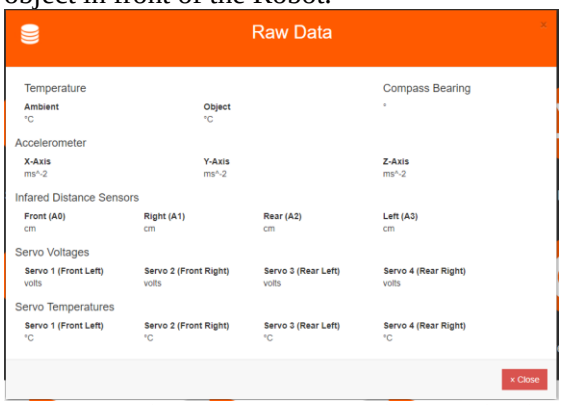


Figure 14 - The Raw Data system shows vital Robot statistics and raw sensor input.

System Logging pulls data from the directory /var/log and displays it in a scrolling window (Figure 15). It is the best way to keep track of events happening on the Robot at an operating system or kernel level.

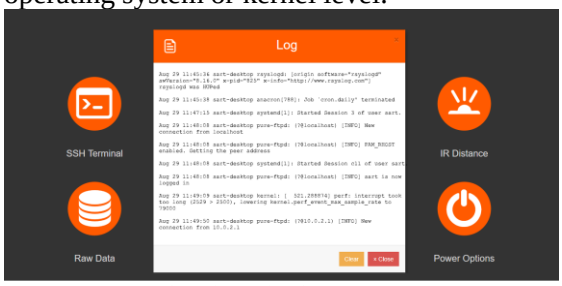


Figure 15 - The log displaying information from /var/log/syslog

FTP File Access, powered by MonstaFTP, allows the operator to quickly edit files on the Robot using an intuitive web interface (Figure 16).

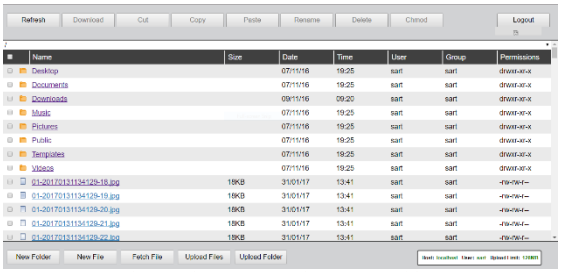


Figure 16 - The Web FTP panel, powered by MonstaFTP.

Power Options let the operator quickly and safely reboot the S.A.R.T Robot, or power it off completely (Figure 17).

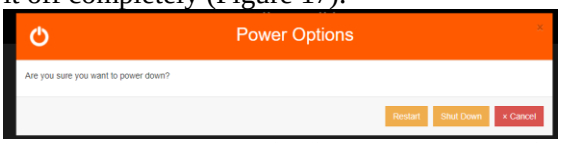


Figure 17 - Power options let the operator turn off the Robot without having to know Linux commands.

The interface dedicates the lower portion of the page to **System Monitoring** (Figure 18). This section keeps track of RAM and CPU usage, CPU temperature, battery statistics and uptime to allow the operator to ensure the S.A.R.T. is functioning normally.

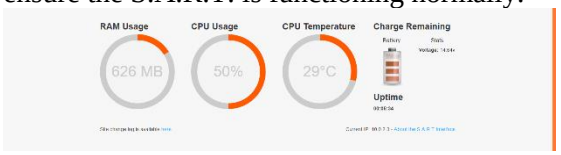


Figure 18 - The lower section of the interface, tracking the NUC and battery statistics.

The **Local Help Documentation** allows the operator to quickly troubleshoot a wide range of possible faults on the fly (Figure 19). This means there is often no need to replace the Control Panel and Robot if something goes wrong, saving valuable time in a rescue situation. All help documentation is local so that operators do not need to research the problem on an internet connected device.

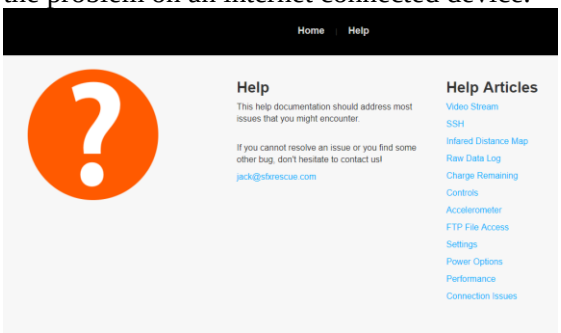


Figure 19 - The main page of the help documentation, featuring an extensive list of articles covering possible faults in each system.

Like the rest of the S.A.R.T. project, the entire interface was made open source during its development to allow other developers to contribute and to implement it in their own projects.

4.0 Application

4.1 Setup of Robot & Operator Station

The setup process of the Robot and Control Panel was designed to be as simple as possible. As mentioned previously, first responders in a rescue situation rate ease of use highly for Robotic assistance. In a high stakes environment, they need something that can go from packed to deployed in a matter of minutes. One of our core design philosophies for user experience was simplicity, meaning the setup process we demonstrated in Japan is remarkably straightforward and intuitive.

Firstly, all devices in the Control Panel are powered on simultaneously with a single, clearly visible power button. The operator can then power on the S.A.R.T. Robot. After the boot sequence, it automatically connects to the waiting S.A.R.T. network. If the operator knows the IP address of the Robot, they can navigate to the Control Panel and start using the Robot immediately. Otherwise, they can use the IP scanner built into the default gateway to find the Robot and continue as usual with minimal time lost.

In Japan, control scripts had to be started manually using the SSH console on the web interface. However, to simplify the process even more and to satisfy the overarching design criteria of an easy-to-use interface that requires minimal prior training to use, these scripts will run automatically on start up in the future.

4.2 Pack Up of Robot & Operator Station

If the operator has already recovered the Robot, the power-off process has three steps. The Robot can be safely shut down using the power options in the interface. The Control Panel can then be turned off in the usual way (Start > Shutdown). Once the device has been safely powered off, power can be cut by depressing the UPS power button. Power can be cut to the Robot using the switch (Figure 20).

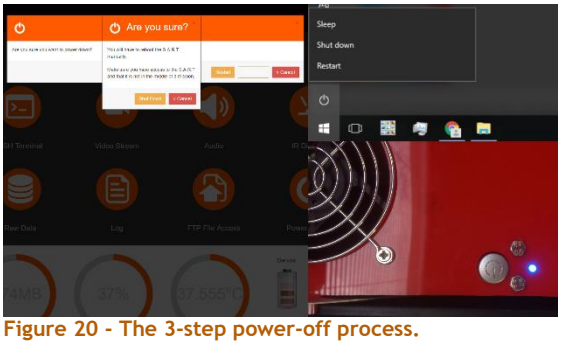


Figure 20 - The 3-step power-off process.

Additional pack up steps may include placing the Robot in its foam lined Pelican carry case, as well as placing a protective foam shield between the monitor and keyboard of the Control Panel before closing the case. Optionally, the battery in the Robot can be removed, although assuming it is not completely flat it is safe to leave it in indefinitely.

4.3 Mission Strategy

The Rapidly Manufactured Rescue League (RMRL) is a sub-division of RoboCup Rescue. It is a research competition that encourages students to develop innovative solutions to open response Robotics problems in Search and Rescue, Hazardous Materials Response, Explosive Ordnance Disposal and Tactical Reconnaissance (oarkit.intelligentRobots.org). The RMRL brings this competition to high schools and undergraduate university classrooms, focussing on the challenges associated with operating small Robots in confined spaces and low-cost prototyping, electronics and sensors to ultimately lower the barrier of entry into Robotics research.

The specific strategy employed by the S.A.R.T. involves a careful analysis of each course to decide on the best route. For example, in the specific courses of *stepfield* and *elevated ramps*, the strategy for navigating was different for the dirt or gravel pits. This strategy involved 'hugging' the left wall (when observed from behind), then executing a 90-degree turn and driving over the elevated section, as opposed to driving diagonally over the elevated section as this would often result in the Robot becoming beached. If this was the resulting eventuality, our solution to becoming unstuck was to rapidly change the direction of the servos, effectively 'rocking' backwards and forwards to build up enough momentum to 'fall' off the perch where the Robot was stuck.

A different strategy was employed for the dirt and gravel pits, because as the Robot is relatively light and the smooth wheel design allowed for the Robot to 'skim' along the top of the surface without sinking in. At the end of each course, the Robot would drive up against the wall and execute a flip instead of turning around, which, in our experience was faster and more efficient, reducing the likelihood of becoming stuck perpendicular to the direction of the course.

These operation decisions were the result of careful experimentation with the courses and from previous experience controlling the robot.

4.4 Experiments & Testing

Much of the experimentation and testing phase occurred early in the development of the physical design and software development. Many of the experiments conducted exposed new problems that had to be solved for the changes made as a result of the experimental findings to be effective.

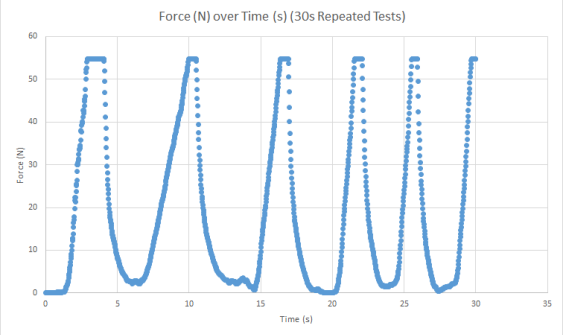
4.4.1 Experimental Design

It was through experimentation that it was concluded that a vertically symmetrical Robot (the same on the top and bottom) would be useful in navigating through confined spaces, as a 180-degree turn was possible by driving up against a wall. Executing one of these turns resulted in the controls swapping and the stream upside down on the operator's end, so we had to patch the Control Interface to include a function that flips the controls and stream footage.

4.4.2 Materials Testing

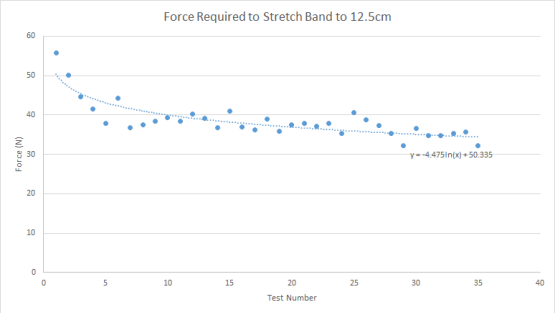
The usefulness of belt prototypes printed in the NinjaFlex flexible 3D printer filament were tested using a Pasco Scientific Force Sensor. This tool was used to measure the force (in Newtons) required to break the flexible filament.

The force sensor only took measurements up to 55 newtons, and the small sample of NinjaFlex did not break, even after multiple tests (Graph 1).



Graph 1 - The results of 6 stretch tests over a period of 30 seconds. The readings stop at 55N as it is the maximum value supported by the sensor.

Additional tests were used to examine other properties of the material, such as how well it retains its flexible elastic properties after stretching (Graph 2).



Graph 2 - Repeatedly stretching the band to 12.5cm to test how elasticity would reduce after repeated strain. Apparent is a moderate negative relationship as the force required to stretch the band to the same length reduced over repeated tests.

4.4.3 Wireless Connection Range

Multiple experiments were run to verify the maximum range of the Xirrus WiFi Access Point. This was done by connecting to the S.A.R.T. network using a laptop (which had the same wireless network adapter as the Robot) and walking in a straight line until the signal was completely lost. The average distance as calculated from the tests was around 90m in a crowded Wi-Fi environment which was thought to accurately simulate the conditions at the competition. Fortunately, this assumption turned out to be correct as we were able to operate wirelessly at ranges of over 50 metres during competition in Japan.

4.4.4 Streaming Reliability

The reliability of streaming video footage was tested by loading the Robot with numerous streaming clients to see at what point the stream cut out or the bandwidth of the Access Point was saturated. Having a low powered dual-core processor in the NUC, the stream began to drop frames, stutter and cut out after more than six clients were connected. The effects of this were exacerbated as the range

increased, but considering only one or two clients would need to be connected to the Robot at any one time, the range and stream quality would remain normal.

4.4.5 Battery Duration Testing

The usable time given by the UPS when the Control Panel is in the 'untethered' mode (i.e. not plugged in) was tested by building the power network required for the Control Panel to function. The worst case scenario was simulated by initiating a synthetic load on the BreezeLite Mini PC and saturating the Access Point with a large file transfer over the Wi-Fi network using FTP. The results gave a minimum usable time of around 75 minutes.
The battery performance of the robot was also tested using a 1300 mAh LiPo battery. A series of tests determined a maximum run time of 90 minutes. We used the information gathered in these tests to ensure enough time was left between runs to swap batteries hourly during competition.

4.4.6 Experimenting at RoboCup

The setup day of the competition in Japan was dedicated to testing the Robot in the competition mazes with the various wheelbase lengths. It was discovered that the long wheelbase tended to get the Robot stuck on flat surfaces in courses such as the *stepfield*, to which we thought that shortening the wheelbase would have a positive effect. After removing everything inside the Robot and reciprocating the orientation of the servos, more tests were conducted with the short wheelbase. It was concluded after a short period of testing that, while the short wheelbase provided better performance in *stepfield*, the performance in every other course decreased. It was therefore decided that the longer wheelbase should be used for the competition as it was the most compatible with the majority of the courses in the competition.

4.5 How the particular strengths of the team are relevant to applications in the field

Each member of the S.A.R.T. has their own defined roles and skill set. The different strengths they possess (Python coding/programming; web developing for front and backend; 3D modelling and rendering; network configuration and administration; building/construction; documentation) can be applied to the field. For example, programming in Python and web development for front and backend ultimately gets the Robot moving and sending the data back for analysis via a web server hosted on the Robot itself. 3D Computer Assisted Design (CAD) and rendering produce the images and simulations of the Robot's use cases, as well as the physical design. On top of all that, a viable network must connect the Robot to the Control Panel despite the difference in hardware and software on each device as created by individual team members. These disciplines require teamwork and communication due to the different people working on the same project having different problem-solving methods different ideas for implementing solutions to the same problem.

This reflects a real world engineering process wherein the team who designs the product may not necessarily be the same team that manufactures the product, who in turn may not be the same team who programs the product or tests the product.

Detailed documentation is necessary in this case, as for the reasons outlined above.
Every member of the team is experienced in documenting their ideas and processes through the website blog to communicate the design and updates through a medium that the entire team can consult readily with ease. It also allows for the easy sharing of multimedia content to explain concepts to not only the team but also the wider open source community.

5.0 Conclusion

5.1 What the team has learned so far

So far through the process of designing and building the many prototypes and iterations of the Robot, the members of the S.A.R.T. have all grown more skilled in their respective areas, leading to more sophisticated and streamlined programs and designs. As a result, we can anticipate potential problems before they occur and amend them before they become a major issue. Specific examples include wireless network optimisation, computer to computer communication via browser based web applications on different operating systems, 3D design and rendering, 3D printing, image manipulation via a remote video stream, controlling servos with a variable speed function using key letter ID numbers sent over a wireless network and building a cheap, rapidly manufactured complete system in a small form factor.

5.2 What is planned between now and the competition in 2018

As we, the current members of the S.A.R.T. are in our final year of college, we plan to find new members who wish to continue the development of this project. However, all existing members have expressed interest in continuing to work on the project, and potentially act as mentors to the students who will take over and continue to improve the project before RoboCup in Canada in 2018.
To aid the continued development of the project, the team has come up with some new ideas for these students to work on in 2018.

5.2.1 Update System

Useful for when the S.A.R.T. has been deployed in the field. This system will notify the operator of a S.A.R.T. Robot when there are software updates available. It should have the ability to automatically download and install the updates if the operator approves the update.

5.2.2 Audio Communication

Set up a streamlined audio communication system using the VoIP protocol rather than two separate text-to-speech and speech-to-text systems. This should allow the operator to talk to a victim directly, much like a phone or Skype conversation.

5.2.3 Browser Cross-Compatibility

Implement support for the S.A.R.T Control Interface on other popular browsers, such as Firefox and Internet Explorer.

5.2.4 Expand the Versatility of the Control Panel

Create a custom battery for the UPS that utilises the space inside the Pelican case more efficiently. With the extra stored power, implement a LiPo charging system into the Control Panel to allow Robot batteries to be recharged straight off the UPS for an increased run time in remote locations or when power is not available.

5.2.5 Dynamic S.A.R.T. Network

Create a network that can be boosted to cover a larger range by multiple Control Panels. Allow multiple Robots to be connected to the dynamic network at the same time, and let each operator select a specific Robot to control. This reflects a real world rescue situation where multiple cheap Robots would be used at the same time to cover more ground.

5.2.6 Design Alterations

Based on the observations of the current S.A.R.T team from RoboCup 2017, the new team can make informed decisions on physical chassis and wheel design changes to better navigate courses.

5.2.7 Change the Main Computational Component

Change the main computational component from a dual-core Intel NUC to the UDOO X86 ULTRA. Although more expensive than the NUC, it features a quad core processor and 8GB of RAM and is capable of running any x86 based operating system, including the Arduino™ 101 world, including all the sketches, libraries and the official Arduino 101 IDE. It has General Purpose Input/Output (GPIO) pins for attaching sensors directly to the board itself without having to have an Arduino Nano as an intermediary between the NUC and the array of sensors.

5.2.8 Migrate to Ubuntu on the Control Panel

The Control Panel is currently using a free trial of Windows Server because the BreezeLite Mini PC only officially supports Windows based operating systems. Unfortunately, the free trial is about to expire, so a different solution needs to be found.

5.2.9 Building a Test Maze

Basing the design off the maze used in Japan, the team could test their Robot at home in the actual competition environment.

The new team in 2018 will bring many new ideas to the table and may choose to implement all or none of these features. These are only our suggestions based on our 3 years of development, testing and competition experience.

6.0 References

Adafruit 2016, *PowerBoost 500 Charger - Rechargeable 5V Lipo USB Boost @ 500mA+*, viewed 19 April 2016, <https://www.adafruit.com/product/1944>.

Cyber Power Systems, Inc. 2017, *CyberPower Value1000EI*, viewed 24 February 2017, <https://www.cyberpower.com/au/en/product/sku/Value1000EI>.

Ewyk, R 2016, *BatView® Sonar Viewer*, viewed 20 May 2016, <https://drive.google.com/file/d/0B06CRDwuKGLLLXBEN0Zyc3JnSzA/view>.

Friends-of-Fritzing foundation 2017, *Fritzing*, viewed 17 July 2017, <http://fritzing.org/home/>.

GizmoSphere 2014, *Gizmo 2 Specifications*, viewed 5 July 2016, <http://www.gizmosphere.org/wp-content/uploads/2014/11/4531_Gizmo2_ProBRIEF_FNL.Element14.pdf>.

Grinberg, M 2013, *How to build and run MJPG-Streamer on the Raspberry Pi*, viewed 3 July 2015, <https://blog.miguelgrinberg.com/post/how-to-build-and-run-mjpg-streamer-on-the-raspberry-pi>.

Gus 2015, *Build a Raspberry Pi Webcam Server in Minutes*, viewed 20 May 2016, <https://pimylifeup.com/raspberry-pi-webcam-server/>.

Intel 2016, *Intel® Centrino® Advanced-N 6205*, viewed 14 September 2016, <http://www.intel.com/content/www/us/en/processors/centrino/centrino-advanced-n-6205-brief.html>.

Intel 2016, *Intel® NUC Board NUC5CPYB and Intel® NUC Board NUC5PPYB Technical Product Specification*, viewed 2 March 2017, <http://www.intel.com/content/dam/support/us/en/documents/boardsandkits/NUC5CPYB_NUC5PPYB_TechProdSpec11.pdf>.

Intel 2016, *Intel® NUC Kit DN2820FYKH*, viewed 9 September 2016, <http://ark.intel.com/products/78953/Intel-NUC-Kit-DN2820FYKH>.

Lavrsen, K 2016, *Motion - Web Home*, viewed 20 May 2016, <http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome>.

Monsta Limited 2016, *Monsta FTP Open Source PHP File Manager*, viewed 20 May 2016, <https://www.monstaftp.com/>.

NinjaTek 2017, *NinjaTek High Performance 3D Printing Materials*, viewed 2 June 2017, <https://ninjatek.com/>.

PASCO Scientific 2017, *Wireless Force Acceleration Sensor*, viewed 11 June 2017, <https://www.pasco.com/prodCatalog/PS/PS-3202_wireless-force-acceleration-sensor/index.cfm>.

PC Case Gear 2016, *BreezeLite SN3-X5 Z8300 Windows 10 Mini PC*, viewed 13 September 2016, <https://www.pccasegear.com/products/36378/breezelite-sn3-x5-z8300-windows-10-mini-pc>.

Popov, D 2014, *PHP on Raspberry Pi*, viewed 20 May 2016, <http://www.raspberry-pi-geek.com/Archive/2014/07/PHP-on-Raspberry-Pi>.

Raspberry Pi Foundation 2016, *Build a LAMP Web Server with WordPress*, viewed 20 May 2016, <https://www.raspberrypi.org/learning/lamp-web-server-with-wordpress/worksheet/>.

RoboCup Federation 2016, *RoboCup 2017*, viewed 9 September 2016, <https://www.robocup2017.org/eng/index.html>.

ROBOTIS Inc 2016, *Dynamixel AX-18A*, viewed 13 September 2016, <http://support.robotis.com/en/product/dynamixel/ax_series/ax-18f.htm>.

Sheh, R 2015, *The Open Academic Robot Kit*, viewed 4 May 2015, <http://oarkit.intelligentrobots.org/wiki/doku.php>.

Sheh, R 2017, *Controlling servos using the USB2AX V3.2*, viewed 3 March 2017, <http://oarkit.intelligentrobots.org/home/raspberry-pi/controlling-servos-using-the-usb2ax-v3-2/>.

Ultimaker B.V. 2017, *Ultimaker Error messages*, viewed 11 May 2017, <https://ultimaker.com/en/resources/160-error-messages>.

Williams, J 2016, *S.A.R.T Interface*, viewed 20 May 2016, <https://drive.google.com/file/d/0B06CRDwuKGLLVVRrei15Wk9jLTg/view>.

Xevelabs 2017, *Product: USB2AX*, viewed 3 March 2017, <http://www.xevelabs.com/doku.php?id=product:usb2ax:usb2ax>.

7.0 Appendix A - Components & Estimated Total Cost

Table 1 - Components & Estimated Total Cost - Robot Only

Component	Price (USD)	Quantity	Total (USD)
Intel NUC5CPYH	$122.36	1	$122.36
Corsair CMSO4GX3M1C1600C11 4GB (1x4GB) DDR3L SODIMM	$39.20	1	$39.20
Samsung PM851 MZ-7TE128D 128GB SSD	$64.00	1	$64.00
Dynamixel AX-18A	$105.52	4	$422.08
oCam 5MP USB 3.0 Camera	$96.16	1	$96.16
Arduino Nano	$28.80	1	$28.80
Infrared Proximity Sensor - Sharp GP2Y0A21YK	$11.16	4	$44.64
Triple-axis Accelerometer+Magnetometer (Compass) Board	$11.96	1	$11.96
MLX90614ESF-AAA Infrared Temperature Sensor	$23.42	1	$23.42
Plasti-Dip	$26.40	2	$52.80
Ultimaker ABS 3D Printer Filament 1kg spool	$32.00	6	$192.00
Quanum 12V-5A (7.2 - 25.2V) Dual Output UBEC	$10.25	1	$10.25
USB2AX v3.2a	$68.56	1	$68.56
0.5m USB 2.0 A male to 5-Pin Mini-B Lead	$4.76	1	$4.76
0.5m USB A male to Micro-B Lead	$4.76	1	$4.76
Turnigy Nano-Tech 1.3 Ah Lithium Polymer Battery	$15.58	3	$46.73
		Total	$1,232.48

Table 2 - Components & Estimated Total Cost - Control Panel Only

Component	Price (USD)	Quantity	Total (USD)
Pelican PROTECTOR CASE™ Camera Case (1514)	$229.66	1	$229.66
BreezeLite SN4-X5 Windows 10 Mini PC	$279.20	1	$279.20
CyberPower Value GP 1000VA/530W [VALUE1000EI] Line Interactive Ups	$138.40	1	$138.40
AOC E2070SWN 19.5in Widescreen LED Monitor	$87.20	1	$87.20
Xirrus XR620 WiFi Access Point (inc. PoE Injector & Ethernet Cables)	$460.00	1	$460.00
Microsoft All-In-One Media Keyboard	$45.60	1	$45.60
HDMI to VGA + Stereo Audio Converter	$22.36	1	$22.36
Allocacoc PowerCube 5 Power Outlets	$22.30	1	$22.30
0.5m VGA Monitor Connecting Cable	$5.56	1	$5.56
Comsol Male IEC-C14 to Female IEC-C13 Power Cable 2m	$11.90	1	$11.90
Comsol Male 3 Pin Plug to Female IEC-C13 Socket 2m	$11.90	1	$11.90
		Total	$1,314.09

8.0 Appendix B - Components

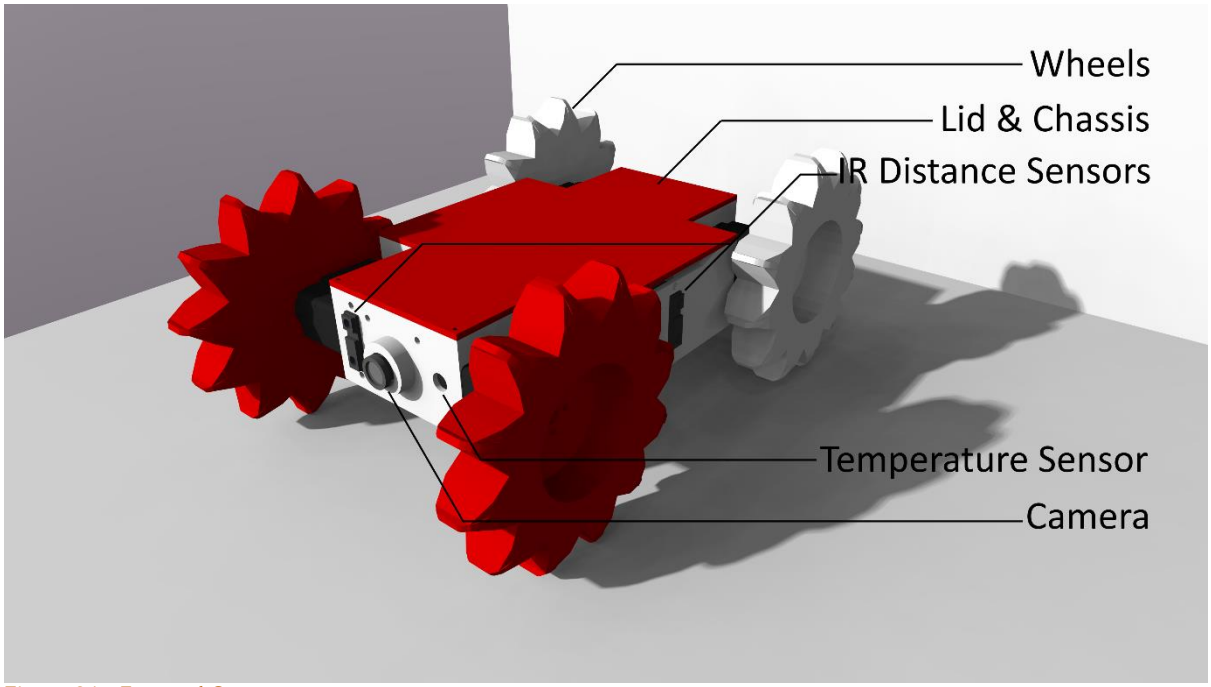


Figure 21 - External Components

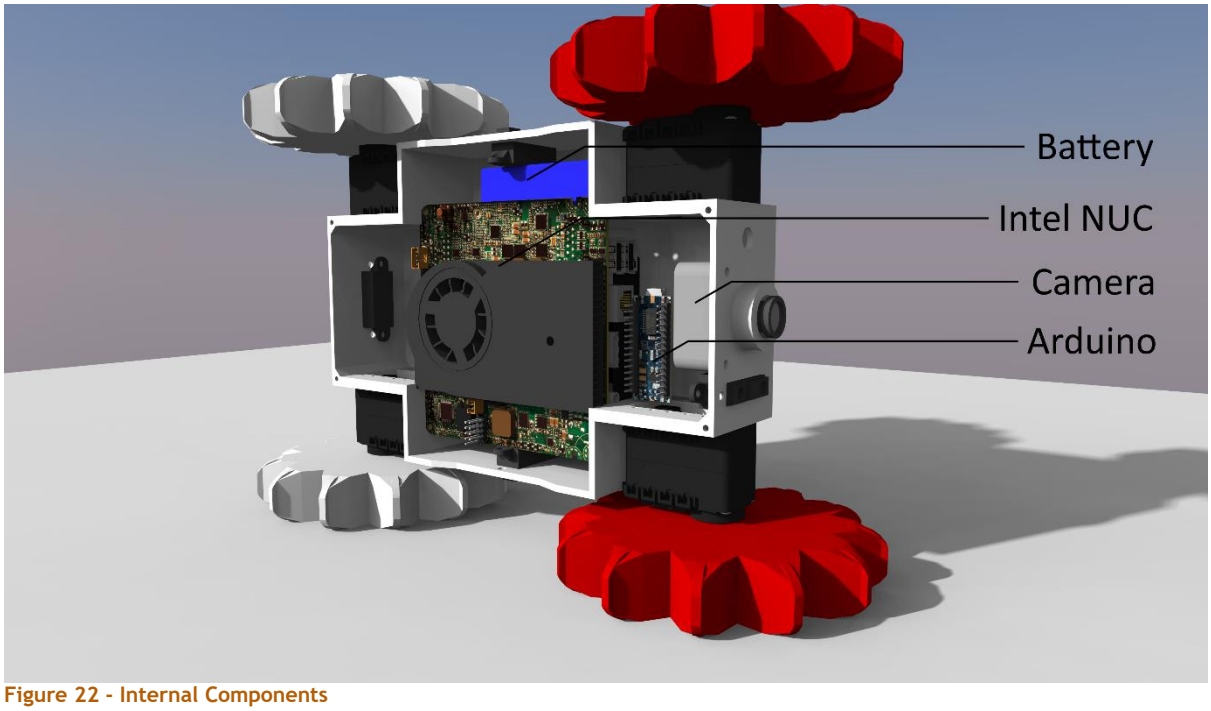


Figure 22 - Internal Components

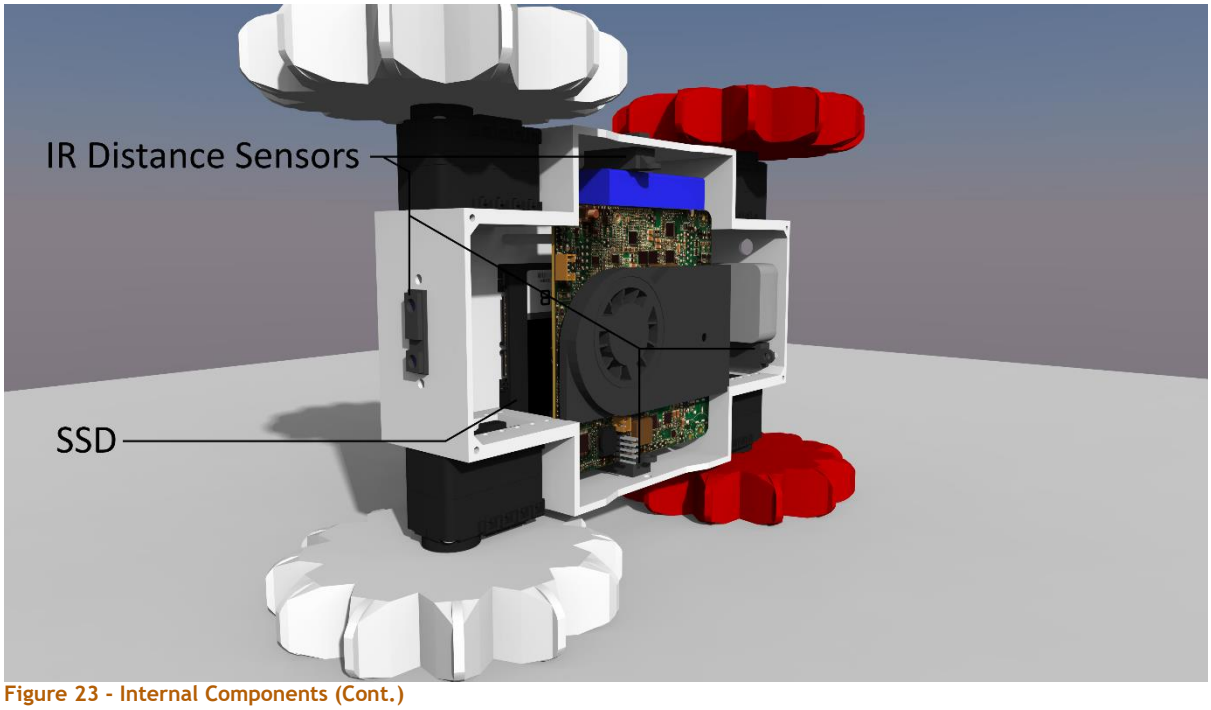


Figure 23 - Internal Components (Cont.)

9.0 Appendix C - List of Software Packages

Table 3 - Software Packages and Dependencies Used

Device or Process	Software Package/s used
BreezeLite Mini PC	Windows Server 2016
	Windows DHCP Server
	FileZilla FTP Client
	Windows Remote Desktop
	CyberPower PowerPanel Personal Edition
	S.A.R.T. Web-Based Control Interface
Intel NUC	Ubuntu 16.04 “Xenial Xerus”
	Motion
	Geany
Movement	Python - Pyax12 - AsycIO - WebSockets
Image Recognition	Python: - OpenCV 3.1.2 - Numpy - Matplotlib
Motion Detection	Python: - OpenCV 3.1.2 - Numpy
QR Code Reading	Python: - OpenCV 3.1.2 - Pyzbar - Imutils - Numpy - Matplotlib
Audio – Text to Speech	Python: - Pydub - WebSockets
Audio – Speech to Text	Python: - speech_recognition - pyaudio - wave - WebSockets
Arduino Nano	Arduino
Robot Chassis	Google SketchUp
Render of Robot & Control Panel	IRender nXt SketchUp Plugin

10.0 Appendix D - List of Hardware

Table 4 - List of Hardware and Chassis Parts

Component	Hardware
Robot Chassis	Heavily modified design inspired by the Emu Mini 2 from the Open Academic Robot Kit.
Robot Wheels	Heavily modified design inspired by the Emu Mini 2 from the Open Academic Robot Kit.
Intel NUC5CPYH	Central computational device on board the Robot.
Corsair CMSO4GX3M1C1600C11 4GB (1x4GB) DDR3L SODIMM	Random Access Memory (RAM) for the Intel NUC.
Samsung PM851 MZ-7TE128D 128GB SSD	Solid State Drive (SSD) for the Intel NUC storage.
Dynamixel AX-18A	Servo.
oCam 5MP USB 3.0 Camera	Camera.
Arduino Nano	Arduino Nano, responsible for collecting all the data from the sensor array and sending it to the Intel NUC.
Infrared Proximity Sensor - Sharp GP2Y0A21YK	Infrared Distance Sensor.
Triple-axis Accelerometer+Magnetometer (Compass) Board	Accelerometer & Compass.
MLX90614ESF-AAA Infrared Temperature Sensor	Infrared Temperature Sensor.
Plasti-Dip	Liquid plastic, used for coating the wheels to make them grip onto the ground or walls.
Ultimaker ABS 3D Printer Filament 1kg spool	The 3D printer filament, a type of plastic extruded through a hot nozzle that builds a model by laying subsequent layers on top of each other.
Quanum 12V-5A (7.2 - 25.2V) Dual Output UBEC	Power delivery for the Intel NUC and servos.
USB2AX v3.2a	A small USB device that allows the servos to interface with the Intel NUC.
0.5m USB 2.0 A male to 5-Pin Mini-B Lead	USB cables and connectors, necessary to connect the Arduino to the Intel NUC for sensor data collection functionality.
0.5m USB A male to Micro-B Lead	USB cables and connectors, necessary to connect the camera to the Intel NUC for video streaming functionality.
Turnigy Nano-Tech 1.3 Ah Lithium Polymer Battery	Powers the entire mobile apparatus.

11.0 Appendix E - Web and Open Source Presence

Under our open source philosophy, everything we did was published online in the form of regular blogs, code repositories and 3D model downloads.

Website Blog

A regular blog detailing the design and implementation process over the course of the 3 year project.
https://www.sfxrescue.com

Code Repositories on GitHub

All our code was made free to use and edit under the GNU-GPL license on *GitHub*, where we encouraged other teams to contribute to or find inspiration in our solutions.
https://github.com/SFXRescue/SARTRobot

Editable 3D Models on Thingiverse

All our 3D models were made free to use and edit under the GNU-GPL license on the 3D model sharing website *Thingiverse*.
https://www.thingiverse.com/SFXRescue/designs

YouTube Channel

The S.A.R.T YouTube channel has a series of videos including tutorials (Plasti-Dipping wheels to improve grip and daisy-chaining the Dynamixel servos), experiments (WiFi range and stream tests) and sharing new features and developments.
https://www.youtube.com/channel/UCOM41hoo5jFGdlgnjjvApSQ/videos