SEMI-AUTONOMOUS RESCUE TEAM

Team Description Materials Q1 2020

Connor Kneebone, Alexander Cavalli, Jack Williams, Matthew Williams, Graham Stock, Charlotte Drury, Anthony Gambale, Michael Cavalli, Nathaniel Kneebone, Martin Hosking, Alexander Thorning

Table of Contents

Semi-Autonomous Rescue Team

Team Description Materials Q1 2020

Logistical Information

Team Name	Semi-Autonomous Rescue Team (S.A.R.T.)
Organisation	St. Francis Xavier College
Country	Australia
Mentor	Graham Stock
Contact Person	Graham Stock Phone: +61 417 439 452 Email: Graham.Stock@sfx.act.edu.au
Website	https://www.sfxrescue.com
Team Email	contact@sfxrescue.com

Figure 1 - The S.A.R.T. (from left to right) Anthony Gambale, Alex Thorning, Martin Hosking, Nathaniel Kneebone Connor Kneebone, Aaron Maggs, Graham Stock, Michael Cavalli, and Alex Cavalli, with the Australian Federal Police Bomb Response Group, 2019.

1 Introduction

The Semi-Autonomous Rescue Team (herein known as the S.A.R.T.) is a group of STEM enthusiasts originally formed in early 2015 with the intent of developing and creating a robot capable of competing in the 2016 Rapidly Manufactured Rescue Competition (RMRC) at RoboCup in Leipzig, Germany. The current team is focusing on modifying the new base design, introduced at the Sydney RoboCup in 2019, to be more compact, functional and reliable.

A major focus of development since last year's competition has been the formal implementation of the SIGHTS software, a complete, open-source teleoperation software suite compatible with most robot hardware.

The team is also in the process of developing a partnership with an industry leader in network communication technologies and look forward to seeing how this partnership will enhance the wireless communication abilities between our robot and control panel.

1.1 Note on the current COVID-19 situation

Due to the current COVID-19 situation, some of our suppliers including Adafruit have temporarily shut down or are due to shut down their operations until further notice. As a result of this, the team has had difficulty sourcing several components for the robot and control panel.

2 Reflection on Sydney, 2019

Moving towards the 2019 competition in Sydney, Australia, the team worked with current industry solutions for rescue robots and disaster-response rovers to develop an all new robot which better suits the RMRC competition testing methods, as well as being a much more marketable solution for emergency services agencies. Inspired by these commercial designs the team participated in RoboCup at Sydney, 2019 with a prototype of the Mark IV. This robot was a major change from our

previous design and represented the biggest jump in technology since the project's inception.

As the whole hardware platform was changed there were bound to be performance and quality issues, especially with components that had not been implemented in earlier designs. For example, during the manipulation test we discovered a major flaw in our robot design, the head was too heavy for the arm to lift. This also severely limited our visual capabilities; the main forward camera was mounted on the head which was at times difficult to control. To rectify this the non-essential components in the head have been moved to the chassis and we have decided to move away from Dynamixel components. The switch to DC motors increases complexity of the hardware needed but still reduces costs and allows us to increase torque. The chassis frame is also being modified to be more manoeuvrable in the tighter spaces of the course.

3 System Description

3.1 Hardware

3.1.1 Robot

3.1.1.1 Chassis Design

The Mark IV is vastly different from its predecessors. It was introduced in Sydney 2019, and has seen a number of changes and modifications to the chassis to fix issues we found during the competition. One of the most significant changes we made when introducing this design is the addition of a manipulator arm.

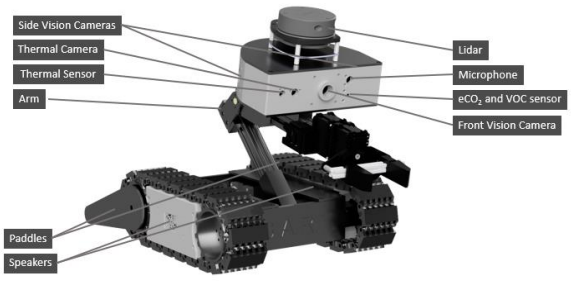

Figure 2 – The prototype S.A.R.T. Mark IV

Instead of having all the components in the body of our robot, as in our previous configurations, the control board and some of the electronics were moved to a separate housing mounted on the robot's arm (Figure 2). As mentioned previously the weight of the head unit was too great for the capacity of our motors. To address this issue, further balancing of components between the two sections is occurring.

One of the main goals of the Mark IV was to improve its marketability with first responders by increasing its practicality as a confined-space rover. With the inclusion of the robot's arm, in conjunction with the rear paddles, the robot has significantly improved manoeuvrability over difficult terrain as well as new capabilities provided by its long reach such as opening doors. Much of the inspiration for this robot's design came from the Servosila "Engineer" Mobile Robot (Figure 3), which is designed to achieve all of the requirements of first responders although comes at a very high price point.

Figure 3 - Example of the long reach capability the Mark IV was designed to achieve. Robot pictured is the Servosila "Engineer". Source: https://www.servosila.com/en/mobile-robots/index.shtml

The head of the Mark IV is in prime position atop the arm to allow the operator to easily see over obstacles when the arm is extended. The expanded vision of this model, achieved through its camera array and LiDAR optimises the robot for confined spaces; no matter the size of the space and orientation of the robot, the operator can always get a picture of what is happening on all sides of the robot.

Changes were made to the dimensions of the chassis since the prototype to allow for extra manoeuvrability in confined spaces.

Further applications of the Mark IV's extendibility include reaching people in confined or difficult to reach locations; the gas sensor and microphone, both contained within the face of the head, can be brought directly to the person. The positioning of the main control electronics on the arm also allows for these electronics to be moved away from environmental factors such as interference or water.

3.1.1.2 Main Computational Board

The Mark IV robot uses a Nvidia Jetson Nano control board, replacing the UDOO x86 Ultra used in the previous model. It was decided that the small form factor and machine-learning capabilities of the Jetson made this a perfect control unit for a small, low-cost rescue robot.

3.1.1.3 Movement

The Mark IV uses two 50:1 Metal Gearmotor DC motors as the main driver of the tracks. The tracks are made of multiple plastic interlocking sections with rubber pads on the top. Each track is connected to the DC motor via a 12-toothed sprocket, and at the other end of the robot, the companion sprocket is mounted on a custom bearing mount through which a shaft is threaded to control the paddles. The two rear paddles are controlled by a Machifit JGY-370 worm gear DC motor mounted to the base of the robot. The robot's arm is controlled by another three of the same motor, which dictate the main movement of the arm, and two micro motors which control the manipulator.

The previous arm had a few issues, particularly to do with the positioning of the cameras and movement capabilities. The positioning of the cameras made it difficult for the driver to determine where the arm was and what position it was in. To fix this, cameras were mounted in better places to give the driver better depth perception and improve their spatial awareness while controlling the robot. The movement capabilities of the robot were limited due to the lack of power that the old Dynamixel servos could produce, the robot would struggle in difficult terrain such as sand.

3.1.1.4 Motor Controller

The switch from Dynamixel servos to DC-motors has necessitated the use of dedicated circuitry to control them. As fine control of multiple motors on one board was necessary, we decided to develop our own board based on the TI-DRV89XX family of drivers, as commercial solutions were either too expensive or not feasible. This is currently in development, but progress has stalled due to difficulty sourcing parts as mentioned in Section 1.1.

3.1.1.5 Sensors

The robot has the following sensors:

- 1x 360° Laser Distance Scanner (LDS-01) – Provides full 360° distance data which can be used to generate maps of the environment.
- 1x CO2 / TVOC Sensor (SGP30) – Measures carbon dioxide and total volatile organic compound levels in the atmosphere. This is used to assist in determining whether a victim is breathing and the dangers of sending in human rescue personnel.
- 1x Thermal IR Camera (MLX90640) – Provides thermal imaging, useful in identifying heat sources like a human body. It is situated on the front of the robot.
- 1x Hall Effect Sensor (AH180) – Used to detect magnetic fields.
- 1x Non-Contact Temperature Sensor (MLX90614) – These can measure the heat of an object up to approximately a metre away.
- Microphone and stereo speaker array – These can be used for two-way communication between the robot's environment and the operator.

3.1.1.6 Vision

The position of the cameras has been radically changed since last year. Previously most of the cameras have been mounted in the chassis walls to provide an approximately 360-degree view from

the robot. This proved to be ineffectual for our operator as many obstacles around the front of the robot were obscured and the angle was unintuitive to steer with.

After meeting with the Australian Federal Police's bomb squad and driving their robots it was decided that only a rear chassis mounted camera was necessary. The other cameras were to be mounted on the head behind the manipulator, as in the last iteration, and on a separate fixed pole on the rear of the robot. This was taken directly from the active bomb disposal robots we used; it provides a third person, over the shoulder view of the robot which is much more intuitive to drive with and provides better obstruction detection than the gripper or chassis mounted designs.

3.1.1.7 Power

The battery in use is a 2200 mAh 25C 4S LiPo. During our testing this provided roughly 120 minutes of continuous active operation. This time is extended by the inclusion of a two-pole rotary on/off switch commonly used on rockets as they can maintain switch position under harsh conditions. The battery outputs at approximately 16V but this can differ depending on the charge level so an UBEC voltage regulator is used to provide a consistent 5V to the main control board and 12V to the motor controller board.

3.1.1.8 Wireless Connectivity

The robot features an Intel AC9260 Wi-Fi card for high speed and high bandwidth wireless communication. This allows us to stream up to four camera feeds to the interface in real-time along with data from an array of sensors.

During the competition in Sydney there were issues involving the antenna wires breaking due to the way they were mounted on the lid. This issue has been fixed by relocating the antennas to the body of the robot.

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 lightweight, portable package that can be easily transported (Figure 4).

Figure 4 - A conceptual render of the S.A.R.T. control panel.

The control panel contains a 15.6" USB monitor which connects to the UDOO x86 Ultra computer. The computer is connected to a Ubiquity Unifi access point, which was chosen for our application based on its high speed and reliability, providing a fast, stable connection even in situations with a considerable amount of wireless interference.

The large and heavy UPS used in previous years was replaced with a much smaller and lighter alternative. An Energizer 500W 12V DC to 230V AC modified sine-wave inverter connected to a 4000mAh 4S Li-Po battery now provides power for the entire control panel. This was an innovative solution to the portable power requirements of a rescue-focused control panel and tests will be performed leading up to and at the competition to assess this solution's effectiveness.

The main panel was made of laser cut acrylic using the Rayjet 300 laser engraver. An additional plate was added to hold up a keyboard with integrated touchpad for interfacing with the control panel. The plate is being held up by 6 small nylon stand-offs placed to support the weight of the keyboard evenly.

On all versions of the control panel, solid-state flash storage is used. This means that there are no moving parts that can be damaged if the control panel experiences 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.

Despite our use of a dedicated control panel, any device with a compatible web browser can be used to operate the S.A.R.T. as long as both robot and controlling device are on the same network. This is because the control interface is hosted on the robot itself via a 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 SIGHTS

The S.A.R.T. Integrated GUI and Host Teleoperation Service (SIGHTS) is the direct result of three years of development work on the original S.A.R.T. Interface. Over this time, the project has evolved to become more modular and encourage collaboration, with the team devoting significant time to supporting other groups using the software through tutorials and instructional blog posts as well as direct support.

S.A.R.T.'s belief in the open source model, an ideology that has been promoted by the team since its inception, has become a resounding success, as evidenced by the involvement of people outside the team in the continued development of SIGHTS.

3.2.1.1 Problem Definition

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 computer experience and prior training.

To accomplish this, SIGHTS is designed to put the operator first based on the input from our team of operators as well as operators from the Australian Federal Police's Bomb Response Squad.

This design philosophy guides the creation and continued development of SIGHTS, not only on the back end where modularity makes adding new sensors or motors trivial, but especially on the front end where end users will be interacting with their robot.

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, modern and intuitive interface.

3.2.1.2 Languages

SIGHTS combines the strengths of a number of popular modern languages.

Python runs on the back-end as a service under Supervisor. Python was chosen because of its versatility, ease of use and popularity. A benefit of this language is the availability of additional Python libraries such as D*ynamixel-SDK*, *asyncIO*, *WebSockets* and *OpenCV* to manage the Dynamixel servos, asynchronous communication operation, control panel communication and vision respectively. Developers extending SIGHTS to work with other hardware can import additional libraries that are easily integrated with SIGHTS due to its modular design (see 3.2.1.4 Modular Design and Extensibility).

JavaScript powers the interface logic. Using some of the more powerful features of the language, SIGHTS grants developers the extensibility they need to implement new ways of representing sensor data (see 3.2.1.4 Modular Design and Extensibility).

3.2.1.3 Features

All SIGHTS configuration is done through a single configuration file which can be edited visually from within the interface, even if the SIGHTS service is stopped or has crashed. Multiple configuration files can be kept and swapped between easily, allowing for easy changes in configuration.

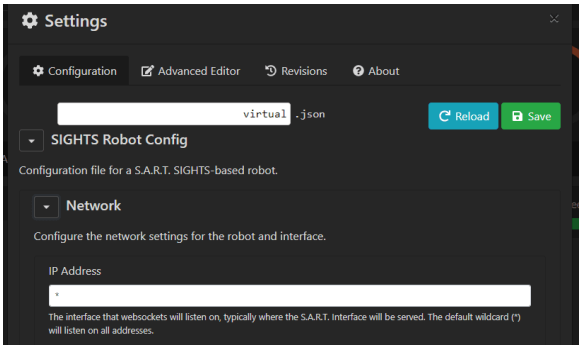

Figure 5. SIGHTS configuration file editor

SIGHTS has a powerful and extensible modular sensor system to allow new sensors to be added with ease. This is enabled by sensor wrapper classes that can use existing Python libraries to access sensors, meaning there is no need to write libraries specifically for SIGHTS. The end user can define which sensors are enabled, where they go on the interface, how they are displayed, and what type of graph they are displayed on. Graphs determine how sensor data is displayed. Sensors can be displayed on multiple graphs, or two sensors can be displayed on the same graph, if the graph supports it.

SIGHTS includes sensor plugins for thermal cameras, ambient and IR temperature sensors, distance (time of flight) sensors, CO2 and TVOC sensors, system memory usage, CPU usage and temperature, system uptime, and disk space usage.

Data from these sensors can be displayed on one of the included graphs including the line graph, percentage circle chart, thermal camera grid or text box.

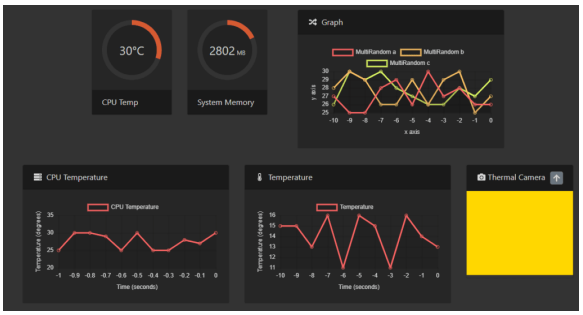

Figure 6. Graphs on the SIGHTS interface displaying data collected by sensor wrappers

SIGHTS has support for adding additional motor connection handlers in a similar manner to sensor wrappers.

On the front end, SIGHTS features an extremely powerful interface that allows the operator to control every aspect of the robot. This interface can have up to four video camera streams through Motion.

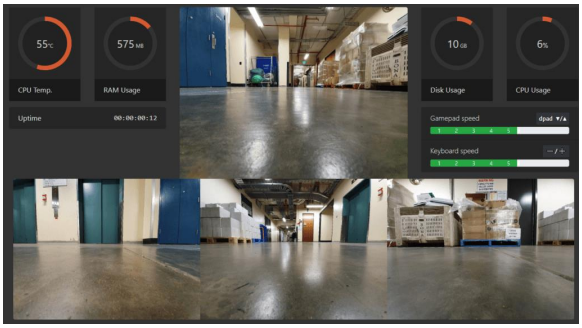

Figure 7. A SIGHTS installation utilising all four camera streams

A tabbed SSH console allows users advanced access to the underlying OS.

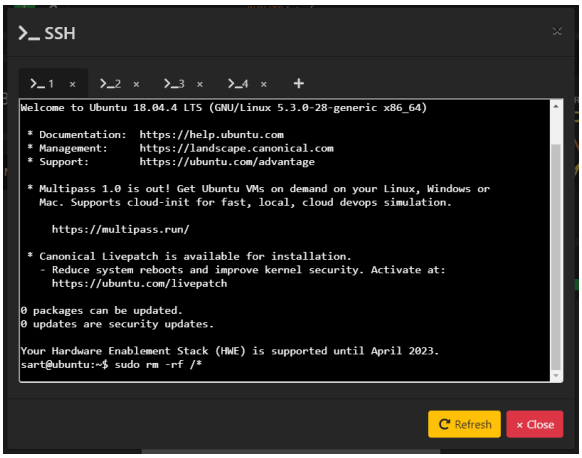

Figure 8. A user running a command via the integrated SSH console

The interface offers light and dark themes to reduce eye strain in high or low light conditions.

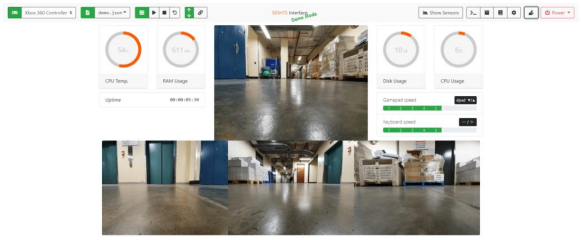

Figure 9. The SIGHTS "light theme"

SIGHTS supports both gamepad and keyboard controls. The operator can use the control scheme best suited to them.

Gamepad support was implemented to make the interface more accessible and easier to learn, inspired by the United States Navy's decision to use Xbox controllers to control the periscopes of their

most advanced submarines [1]. It additionally allows the use of analogue controls, notably the thumb-sticks on the controller. This allows the robot's speed to be controlled with far more precision than with the keyboard.

SIGHTS will feature customisable control configurations, similar to many video games. This is in line with SIGHTS's philosophy of operator-first design, improving accessibility and reducing the need for training.

The specifics of gamepad controls were developed based on input from the team's operators and from discussion with operators from the Australian Federal Police's Bomb Response Squad.

In addition to the visual configuration file editor mentioned earlier, SIGHTS has a text-based editor for power users. The end user can create unlimited config files and swap them at runtime. A config backup is created every time it is overwritten. End users can easily review and restore a previous config version in the revisions tab.

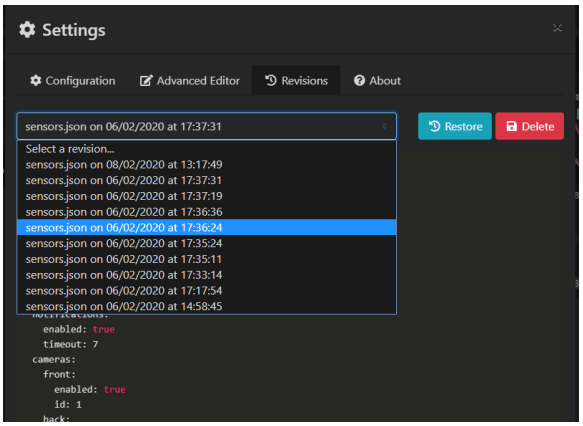

Figure 10. Reviewing an older version of a config file

All config creation, editing, and management features can be used even if the SIGHTS service is stopped or has crashed. This means that a configuration error cannot cause any permanent problems in the software.

To complete the feature set, the operator has the ability to safely shut down or restart the robot through the interface. This means that, under normal use, the operator never needs to access the underlying OS outside of SIGHTS.

3.2.1.4 Modular Design and Extensibility

The SIGHTS software suite is designed to be very extensible when it comes to adding new sensors. This is achieved using Python classes called sensor wrappers to collect sensor data. A sensor wrapper extends the SensorWrapper class and can import any other library to do the heavy lifting – the idea is that you don't need to write libraries specifically for SIGHTS.

JavaScript classes called Graphs are used to display the sensor data received from the host. Graphs are data agnostic - they do not have to be specifically implemented for a certain sensor type. This means an end user can use (for example) the line graph to display anything from temperature readings to CO2 levels over time.

Developers can also implement motor handlers to add support for different servos or motor controllers. This plugin system works just like the sensor plugin system, where a new motor handler extends the Python class MotorWrapper. Just like sensor wrappers, a motor handler can import any external library to connect to the motors.

The team is very supportive of external groups using the software and have written extensive tutorials showing how to create new sensor wrappers for different hardware, and new graphs to display data in different ways.

3.2.1.5 Open Source Presence

The S.A.R.T. has long been recognised by their contributions to the open source community, remaining the only team to have received an Open Source and Innovation Award. With the release of SIGHTS, the team hopes to further encourage collaboration between teams in the Rapidly Manufactured Rescue Challenge and industry experts in the area of small form factor rescue and reconnaissance robotics.

This is promoted through the team's extensive use of GitHub [2] and, at the time of writing, five detailed tutorials explaining how to use and extend the software.

3.2.2 Robot (Host)

3.2.2.1 Operating System

The Nvidia Jetson Nano uses an Ubuntu-based Linux distribution called JetPack. All the software is platform and distribution agnostic, allowing others to easily replicate our project using freely available operating systems.

3.2.2.2 Further Development Projects

There are a small number of development projects separate to SIGHTS. These are usually experimental, or handle edge cases encountered by the team, and thus may not yet be open source.

3.2.2.2.1 Pynamixel Library

With the inclusion of the arm which uses a different series of actuators, a custom *DynamixelSDK*-based Python library was written which serves as an alternative to *pyax12*. It has support for controlling groups of motors across models and protocols.

3.2.2.2.2 Autonomy

The current iteration of the autonomous functionality is a combination of a PID controller and a finite state machine. The PID controller keeps the robot centred in the field, controlling the ratio of power to the left and right wheels while moving forward, and the state machine handles the intersections and ends of the fields. Currently, this only works on the most basic of fields, so the next iterations should work on some of the easier courses with obstacles, and eventually be stable enough to utilise SLAM.

auto.py allows the robot to run in an experimental autonomous mode. It's in very early stages currently but in the future, it will allow the robot to navigate any of the main courses autonomously.

3.2.2.2.3 SLAM

The team is in the process of integrating elements of the Robot Operating System (ROS) project into the robot to incorporate Simultaneous Localisation and Mapping (SLAM). SLAM algorithms combine data from various sensors (e.g. LIDAR, IMU and cameras) to simultaneously compute the position of the robot and a map of the robot's surroundings. SLAM is most commonly used currently in autonomous cars, and the team has had some industry discussion with Bosch on their use of SLAM in autonomous vehicles, though their research is not open-source at this stage.

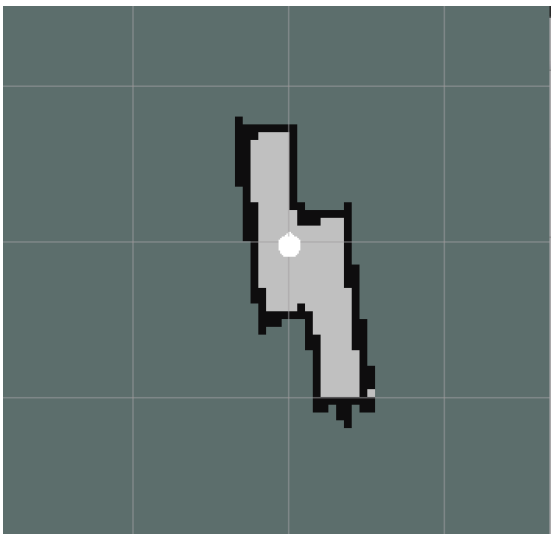

Figure 11 - Map generated through SLAM using LiDAR sensor

ROS provides SLAM-related software that allows use of programs such as *rviz* and *gmapping*. Through the use of a LDS-01 Lidar sensor from Robotis, SLAM will enable the robot to make a 2D map of the surrounding area (Figure 11) and allow it to create a path it can follow autonomously. The map will be seen in real time on the control panel which, in conjunction with the cameras, will allow rescuers to visualise the area and terrain to better assess the situation before personnel must enter.

To further develop our localisation and mapping, we have also begun experimenting with Google Cartographer, an open source real-time SLAM library in 2D and 3D with ROS integration, at the competition in Sydney.

3.2.3 Vision

The S.A.R.T. robot has multiple capabilities regarding interpretation of its surroundings, including recognition of Hazmat warning signs, tracking of moving objects and reading of QR codes (Figure 12). Optimizing the speed and accuracy of the process with new hardware,

algorithms and methods will come afterwards, in preparation for the competition.

Figure 12 - QR code reading

For the hazmat detection, contour detection is utilised to locate the signs (Figure 13). The canny method of contour detection is used as it is the most effective. In order to make the detection more efficient, an image pyramid and a sliding window is used. The image pyramid helps to make larger contours that branch across large sections of the image easier to see, by scaling them down (otherwise they may have imperfections in their edges that the canny algorithm does not recognise).

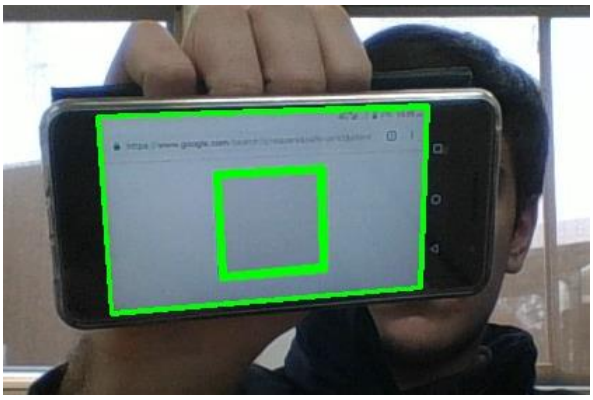

Figure 13 - Canny edge detection displaying rectangular contours only

The sliding window helps us to detect more contours, because it looks in more detail at each part of the image instead of the entire image in one shot.

Different features of the proposed regions, including colour histograms, are extracted and compared to the features of images in a set of training data using the k-nearest-neighbours (*knn*) algorithm. A lot of this will be using the *scipy* library's *knn* implementation to classify the hazmat sign.

3.3 Network Configuration

All communication is done over a wireless network using open-source web protocols. It uses the IEEE 802.11ac 5Ghz standard, as required by the competition. The access point in the control panel is used to allow the robot to connect completely wirelessly, even in high traffic environments with minimal latency and interference.

The basic communication settings used during the 2019 competition was as follows:

Band: 5Ghz
Wi-Fi Mode: A
Channel: Set by competition regulations
Channel bonding: Disabled

The network is configured to use the 10.0.0.x subnet of addresses. The access point is statically assigned to 10.0.0.1 and addresses from 10.0.0.3 – 10.0.0.10 are reserved for each robot. The remaining addresses are part of a DHCP pool and are used for any other device on the network, such as the control panel.

The advantage of using an industrial grade access point over consumer-grade access point, as would commonly be used by households, is that all the bandwidth can be concentrated on a single client. Meaning that there is a reduced chance of experiencing an outage. The reliability this would provide in a rescue situation cannot be understated, as being able to maintain a strong signal is essential and the ability to channel bond and filter out other networks will greatly assist in this.

4 Operational Procedures

4.1 Setup

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 amongst the requirements of a rescue robot. Thus, one of our core design philosophies for user experience was

simplicity, meaning the setup process is remarkably straightforward and intuitive.

Setting up the control panel is simply a matter of opening the lid and pressing the clearly visible power button with LED indicator, provided the LiPo battery is installed. The computer automatically loads the control interface.

The operator can then power on the robot by turning the rotary switch, which then automatically connects to the waiting wireless network.

4.2 Mission Strategy

During operation, the teleoperator has primary control of the robot using their choice of a keyboard or any Gamepad compatible controller, with easy switching between robot driving and arm operation with a single button press.

The teleoperator's primary vision for navigation will be the optical cameras however the suite of other sensors including map of the environment are displayed on the control panel so that the teleoperator has all the information necessary to control the robot.

4.3 Pack Up

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 in desktop operating systems. Once the device has been safely powered off, power can be cut by depressing the control panel power button. Power can be cut to the robot using the rotary switch. Additional pack up steps may include placing the robot in its foam-lined hard carry 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.

5 Experiments & Testing

5.1 Experimental Design

Throughout the leadup to and during the competition the effectiveness of the Mark IV robot design was assessed, focusing on the specific goals this design was meant to achieve. This was done through testing methods which were manufactured here at school as well as those at the competition. In particular, we focused on the manipulation capabilities of this robot as the S.A.R.T. has never deployed a robot with manipulation capabilities. This was done through a 3D-printed pipe-star which we manufactured prior to the competition.

5.2 Wireless Connection Range

The Mark IV version of the robot, in conjunction with the control panel, deploy new communication technologies which have not previously been used. This includes the new access point and significantly improved wireless antenna on the robot. At the competition we were able to test the effectiveness of this new system in the high-traffic environment that the competition provides. Our main issues were reliability with the antennas on the robot themselves, as the cables were prone to breaking.

5.3 Battery Duration Testing

As the Mark IV robot and modified control panel both use new hardware and batteries, continued monitoring and testing will take place to measure the run time of both the robot and the control panel.

6 Future Developments

Future developments will continue to focus on vision processing and autonomy. This will include taking advantage of our new hardware's capabilities in machine learning and vision processing to improve our autonomous functionality and computer vision processing. In future we would like to put significant development into a new innovative control interface, utilising virtual and augmented reality technologies to provide a significantly improved interface between the teleoperator and the robot.

7 Conclusion

The S.A.R.T. Mark IV robot represents a rethink of the team's approach to rescue robots by taking feedback from industry professionals and current market solutions for small-scale rescue robots. At the competition in France, 2020, our overall goal is to assess the performance of these improvements to our new robot design against the competition's tests. The team hopes to achieve significant

developments and improvements over the course of the competition days to give us clear direction moving forward with this new robot design.

Through our conversations with industry experts and research on current solutions for rescue robots we have made a number of strides in our robot design and have learnt a lot about what the industry needs in a rescue robot. Our collaboration has included workshopping days with current and alumni team members, as well as industry, and the success of these days have taught the team the importance of the knowledge and skills of these people bring to our robot development.

8 References

[1] B. Vergakis, "The U.S. Navy's most advanced submarines will soon be using Xbox controllers," 15 September 2017. [Online]. Available: https://www.dailypress.com/military/dp-nws-navy-subs-xbox-controller-0122-story.html. [Accessed 6 March 2019].

[2] S.A.R.T., "S.A.R.T. on GitHub," 2020. [Online]. Available: https://github.com/SFXRescue/. [Accessed 4 March 2020].

[3] Open Source Robotics Foundation, "Turtlebot," 2018. [Online]. Available: https://www.turtlebot.com. [Accessed 20 December 2018].

[4] Servosila, "Mobile Robot Servosila "Engineer"," n.d.. [Online]. Available: https://www.servosila.com/en/mobile-robots/index.shtml. [Accessed 18 June 2019].

[5] C. Kneebone, M. Williams, J. Williams, A. Cavalli, A. Maggs, R. Ewyk, R. Cockerill, M. Cavalli, C. Drury, G. Stock and R. Millard-Cartwright, "S.A.R.T. Team Description Materials for Q1 2019," 17 March 2019. [Online]. Available: https://www.sfxrescue.com/developmentupdates/sart-team-description-materials-for-q1-2019/. [Accessed 3 March 2020].

[6] M. Williams, J. Williams, A. Maggs, R. Ewyk, R. Cockerill, C. Kneebone, B. Hallett, E. Biskup, K. Ewyk and J. Spandler, "Updated TDM for 2018," 18 March 2018. [Online]. Available: https://www.sfxrescue.com/developmentupdates/updated-tdm-for-2018/. [Accessed 4 March 2020].

[7] M. Williams, J. Williams, A. Maggs, R. Ewyk and R. Cockerill, "S.A.R.T. Team Description Materials," 1 September 2017. [Online]. Available: https://www.sfxrescue.com/developmentupdates/sart-team-description-materials/. [Accessed 1 March 2020].

[8] C. Kneebone, A. Cavalli, G. Stock, C. Drury, A. Gambale, M. Cavalli, N. Kneebone and M. Hosking, "S.A.R.T. Team Description Materials for Q2 2019," 19 June 2019. [Online]. Available: https://www.sfxrescue.com/developmentupdates/sart-team-description-materials-for-q2-2019/. [Accessed 3 March 2020].

9 Appendices

9.1 Appendix A – History of SART

After forming in early 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.

Between the 2016 and 2017 competitions we worked to address some of the major issues with the robot that occurred in Germany. This included improved Wi-Fi networking utilising an industrial-grade access point (AP) partnered with an Intel NUC mini PC and a UPS to make up a custom portable control panel. As well as this, we made improvements to the software side of the robot's operation, including real-time motion tracking and image recognition as well as more accurate temperature and distance measurement.

The founding members graduated in 2017, passing the team on to a second generation.

In 2018, the new S.A.R.T. decided to redesign the robot from scratch. This included the chassis, the wheels and all the software written for the project. The previous robot had issues in Japan where it would beach on parts of the course due to the depth of the chassis. One of the first changes that was made was to make the new chassis far shallower. The new board, the UDOO x86 Ultra allowed for this, as it wasn't as tall as the Intel NUC.

The decision was made to experiment with using four cameras instead of the one. This ended up being an excellent choice, providing the navigator with an almost 360-degree view, making the robot far easier to drive. There were also a number of new additions to the sensors. Laser time-of-flight distance sensors replaced the previous infrared (IR) ones, three non-contact temperature sensors instead of one, an all-new thermal camera, and an air quality sensor that measures estimated CO_2 and Total Volatile Organic Compound (TVOC) levels.

The interface was completely redesigned. The original modal-based interface worked well for the old robot, with minimal sensors and only one camera, but adapting that to the current robot with its four cameras was not going to be the most operator-focused system. The new interface was built around the four cameras, which take up the majority of the screen. System information is now integrated into each side of the screen. The three additional cameras (left, right and back) can be swapped at the press of a button for the sensor display which contains the distance sensor graph, the temperature graph and the thermal camera display.

This interface and control service, although basic at the time, eventually became SIGHTS.

A major change in 2018 was the wheel design. The team researched, designed and made polyurethane tyres for the wheels. This replaced the original Plasti-Dip method, which was cumbersome to do, and the material was no longer in production.

Since the project's inception the robot has gone through several iterations, with the competition robots as follows:

- 2016, Germany – Emu Mini 2-based robot with Raspberry Pi B+ and Dynamixel AX-12A servo motors.
- 2017, Japan – Fully customised 3D-printed chassis paired with Intel NUC and oCam 5MP camera for improved video streaming. Control panel was developed with enterprise-grade Wi-Fi as a standalone system with an Uninterruptable Power Supply (UPS).
- 2018, Canada – Re-engineered chassis with UDOO x86 Ultra as the main control board, custom moulded polyurethane wheels, and a

multitude of sensors including laser distance sensors, gas sensors and thermal camera.

- 2019, Sydney – A completely new prototype design, now featuring an arm and a completely new array of sensors.

With the graduation of the second S.A.R.T., the robot was again passed on to a new team. This time, the team consists of students from across Years 9 to 12. With the new team comes a huge range of opportunities for innovation and improvement for the robot.

Overall, the S.A.R.T. have experienced great success in competition, having received awards and recognition for several achievements and advances, 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
- Open Source and Innovation Award – Rapidly Manufactured Rescue League, RoboCup 2018, Montréal, Canada

Throughout the course of our project's development, we have utilised communication strategies such as Open Academic Robot Kit Google Group as well as our own self-deployed communication platform through Mattermost. In particular, several present and past team members have been active participants on the Google Group to build our team identity and strengthen networking within the RMRC community.

9.2 Appendix B - Components & Estimated Total Cost for Mark IV

Note: All costs are based on recommended retail price (RRP) and are in U.S. dollars.

9.2.1 Robot

Component	Price (USD)	Quantity	Total (USD)
Nvidia Jetson Nano	$ 99.00	1	$ 99.00
MLX90640 IR Camera	$ 66.72	1	$ 66.72
ELP 5MP HD USB Autofocus Camera	$ 56.23	3	$ 168.69
LDS-01 LiDAR	$ 171.76	1	$ 171.76
Electret Microphone Amplifier - MAX9814 with AGC	$ 7.95	1	$ 7.95
Adafruit I2S 3W Class D Amplifier Breakout	$ 5.95	1	$ 5.95
Generic Stereo Enclosed Speaker - 3W 8Ω	$ 3.66	2	$ 7.32
Adafruit SGP30 CO2 / TVOC Sensor	$ 19.95	1	$ 19.95
MLX90614 Infrared Temperature Sensor	$ 7.57	1	$ 7.57
Antenna	$ 10.95	1	$ 10.95
50:1 Metal Gearmotor 37Dx52Lmm with encoder	$ 34.95	4	$ 139.80
Machifit JGY-370 DC 12V	$ 16.07	5	$ 80.35
Micro motor	$ 5.00	2	$ 10.00
2-pole rotary switch	$ 6.16	1	$ 6.16
Cables	$ 17.11	1	$ 17.11
Track	$ 25.11	3	$ 75.33
12 Tooth Sprocket	$ 9.95	4	$ 39.80
Paddle Hub	$ 7.65	1	$ 7.65
Servo Tubing Connector	$ 6.95	1	$ 6.95
Carbon Fibre Tube (1m x 24mm)	$ 30.00	1	$ 30.00
Carbon Fibre Sheet (3x400x300mm)	$ 48.59	1	$ 48.59
Carbon Fibre Sheet (4x200x300mm)	$ 45.70	1	$ 45.70
Gripper	$ 40.00	1	$ 40.00
Turnigy 2200 mAh 3S 25C Lipo Pack	$ 12.87	2	$ 25.74
Generic 4 Port USB 2.0 Hub	$ 8.50	1	$ 8.50
Intel AC9260NGW Wi-Fi Card	$ 27.95	1	$ 27.95
Ultimaker ABS 3D Printer Filament 1kg spool	$ 32.00	1	$ 32.00
Quanum 12V-5A (7.2 - 25.2V) Dual Output UBEC	$ 10.25	1	$ 10.25
DRV8912-Q1	$ 5.30	2	$ 10.60
LS7366R	$ 6.79	8	$ 54.32
ACH-14.31818MHZ-EK	$ 2.62	8	$ 20.96
296-18509-1-ND	$ 1.00	2	$ 2.00
Atmega1284P	$ 9.01	1	$ 9.01
AH180 Hall Effect Switch	$ 4.95	1	$ 4.95
Total			**$1,319.58**

9.2.2 Control Panel

Component	Price (USD)	Quantity	Total (USD)
SE830 Waterproof Protective Case	$87.00	1	$87.00
UDOO x86 Ultra	$267.00	1	$267.00
Energizer AC Sine-Wave Inverter	$50.64	1	$50.64
AOC E1659FWUX USB Monitor	$92.38	1	$92.38
Ubiquity Unifi AP AC Pro Access Point	$148.84	1	$148.84
Power-Over-Ethernet Injector	$8.00	1	$8.00
Microsoft All-In-One Media Keyboard	$45.60	1	$45.60
Turnigy 4000mAh 4S LiPo Battery	$36.95	2	$73.90
Xbox Controller	$37.64	1	$37.64
Headset	$20.53	1	$20.53
Acrylic Sheet (3mm)	$20.00	1	$20.00
Power Button	$3.00	1	$3.00
Total			**$855.00**

9.3 Appendix C – Robot Render and Components

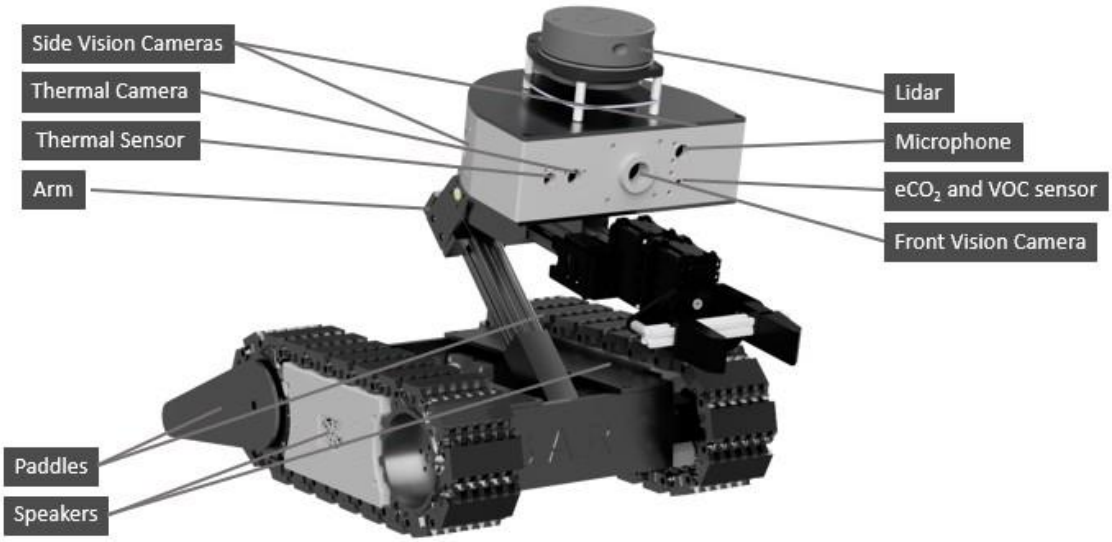

Figure 14 - Render of the Mark IV robot, detailing external components.

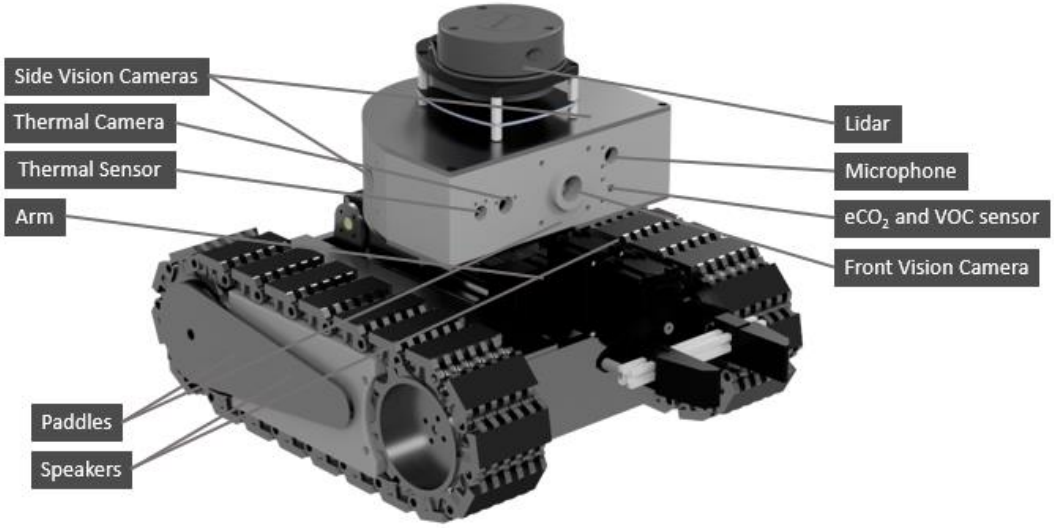

Figure 15 - Mark IV robot compact configuration

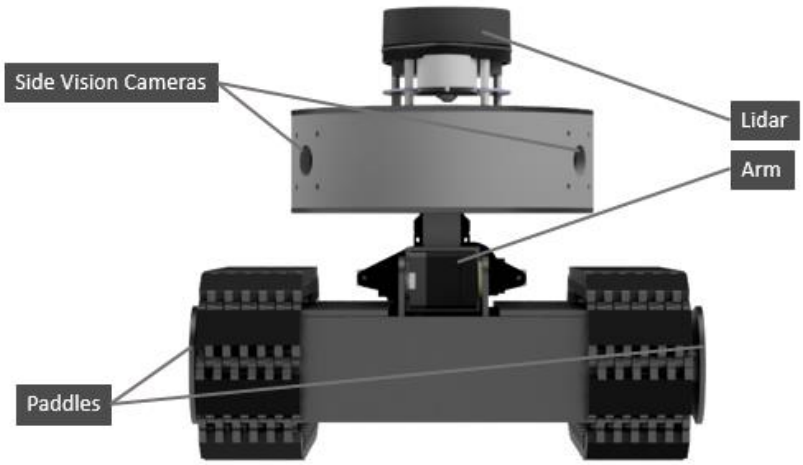

Figure 16 - Mark IV rear view

9.4 Appendix D – List of Software Packages

Device	**Software used**
Control Panel	Ubuntu
	Mozilla Firefox
	Python 3.x
Robot	Ubuntu / Nvidia JetPack
	SIGHTS
	Robot Operating System (ROS)

Process	**Software packages / libraries used**
Complete operation of the robot	SIGHTS
Motion detection, image recognition and QR code reading	Python: • OpenCV 3.1.2 • Pyzbar • Imutils • Numpy • Matplotlib
SLAM	ROS: • rviz • gmapping C++: • Google Cartographer
3D design and render of robot parts	Autodesk Fusion 360 Adobe Illustrator

9.5 Appendix E – Web and Open Source Presence

The S.A.R.T. has long been recognised by their contributions to the open source community, remaining the only team to have received an Open Source and Innovation Award throughout the history of the RMRC.

Under our open source philosophy, everything we do is published online in the form of regular blogs, continual code repository updates and 3D model downloads. The S.A.R.T. continues to encourage other teams to share their own resources in the interest of furthering the development of small form factor rescue and reconnaissance robotics.

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

Code Repositories on GitHub
All our code is provided free to use and edit under the GNU General Public License on GitHub, where we encourage other teams to use, contribute to and/or find inspiration in our solutions.
https://github.com/SFXRescue/

Editable 3D Models on Thingiverse
All our 3D models were made free to use and edit under the GNU General Public 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, experiments and sharing new features and developments.
https://www.youtube.com/channel/UCOM41hoo5jFGdlgnjjvApSQ/videos