CG1112 Engineering Principle and Practice
Semester 2 2017/2018

"Alex to the Rescue"
Final Report
Team: 02-02-01

Name	Student #	Main Role
KARAN SARAT	A0189747E	Team Leader
KEVIN PUTERA	A0184525B	Programming Lead
LEE WEI HERN JASON	A0182856R	Member
SCOTT FLOYD ZHEN-YAN SMAHON	A0183862W	Member
JERRY ZHANG ZHUORAN	A0187394M	Member

Section 1 Introduction

Alex is equipped with a LIDAR(Light Detection and Ranging) device, a system that uses light from a laser and measures the reflected pulses to infer two-dimensional depths, and a rotary encoder on each wheel. These devices will be used as the inputs to a SLAM(Simultaneous Localisation And Mapping) algorithm in order to map the surroundings of Alex and to determine the current coordinates of Alex. Using this map data, the operator will be able to move Alex to unexplored locations to eventually map the whole environment. On the remote client, there will be a UI displaying the map data relayed over from Alex as well as commands that can be sent to Alex.

Alex will be remotely controlled, much similar to an actual search and rescue robot. Commands will be sent to a program running on the Raspberry Pi in Alex over TCP/IP (Transmission Control Protocol/Internet Protocol), which will be interpreted and executed. These commands will include, but not limited to, movement-based commands and object identification commands. These commands will be sent via a remote client that will be connected on the same network as Alex.

Alex will be capable of forward, backward, left and right movement, which will be driven by an algorithm controlling the PWM for each wheel motor. In order to ensure the wheels are moving at the same speed regardless of the level of the battery in Alex, dynamic PWM calculation will be required.

Alex will be able to determine objects within the room and possibly the colour of it as well through the usage of a PiCamera. Alex will be able to determine the colour of the obstacle and relay it back to the operator for recording purposes. This can be done through the usage of PiCamera’s API. [1]

Section 2 Review of State of the Art

Review of state of the art is about reviewing the highest and most recent stage in the development of a relevant product to our project so that we are able to make references to it whenever we face an obstacle [2].

Robot 1: THE HUMANOID ROBOT WALK-MAN

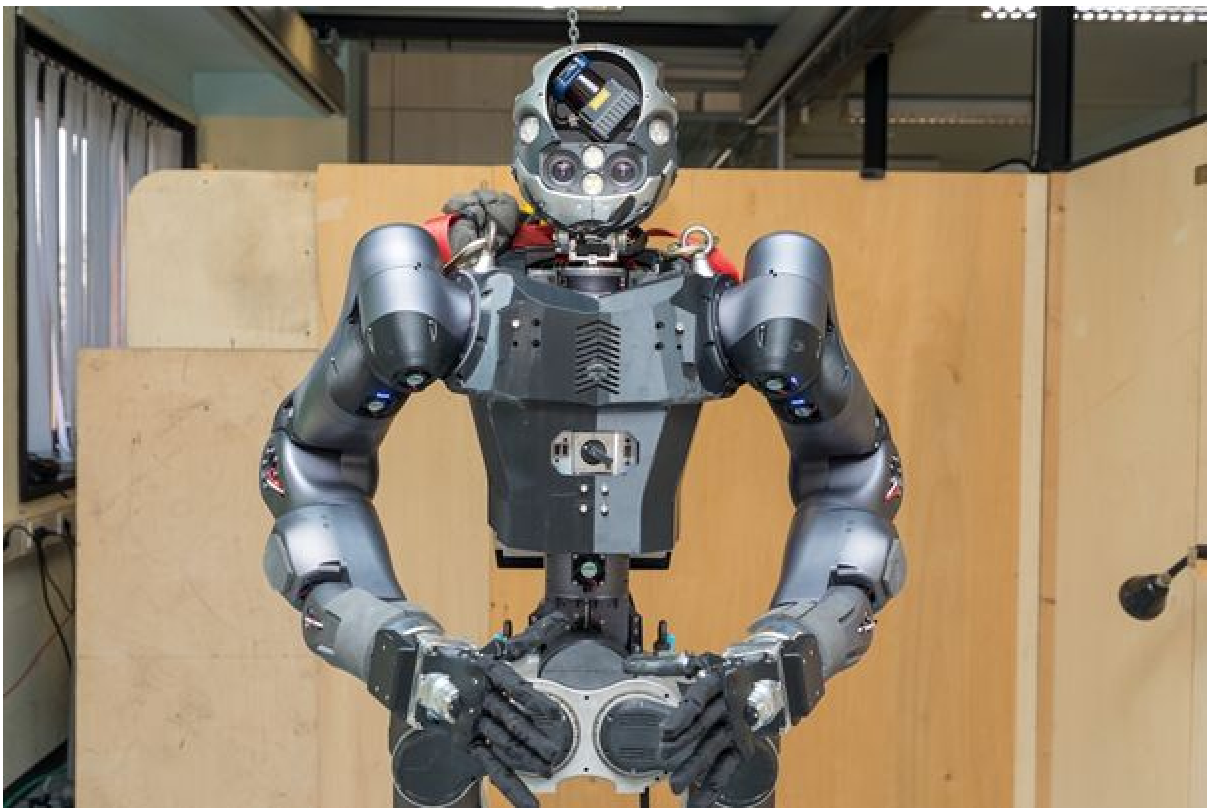


Figure 1. WALK-MAN [3]

The Walk-Man is a custom-made humanoid disaster robot which can operate over uneven terrain and in emergency situations assist or replace humans. It uses a stereo camera and a rotating 3-dimensional laser scanner to map out the environment [4]. In addition to that, it is equipped with microphone arrays, force sensors, and IMUs (Inertial Measurement Unit) to sense the surrounding environment[4]. On the software side, XBotCore framework, YARP middleware, and ROS Gazebo are used. YARP middleware is utilised as the backbone of the communication between modules, while ROS Gazebo is used for high-level operator GUI and 3D perception[4].

All in all, the Walk-Man is a robot that demonstrates human type locomotion, balance, and manipulation capabilities. Its humanoid figure enables it to be more versatile in many cases than other search-and-rescue bots. However, it can only last for 2 hours, which is very troublesome for the user during a search and rescue mission which may require an extensive amount of time [5].

Robot 2: SmokeBot
This robot is a fire fighting robot meant to help fire brigades in search and rescue operations where there are limited visibility and hazardous environmental conditions. As smoke-filled environment impairs visual mapping devices such as LIDARs and cameras, it employs a series of sensors to form the RGT-V unit (radar, gas sensors, thermal cameras-vision) to map the surroundings [6]. Moving on to the software components of the SmokeBot, multiple inputs are considered at the same time, forming the General Disaster Model [6] and this model allows the robot to automatically change according to the hazard level in the environment.

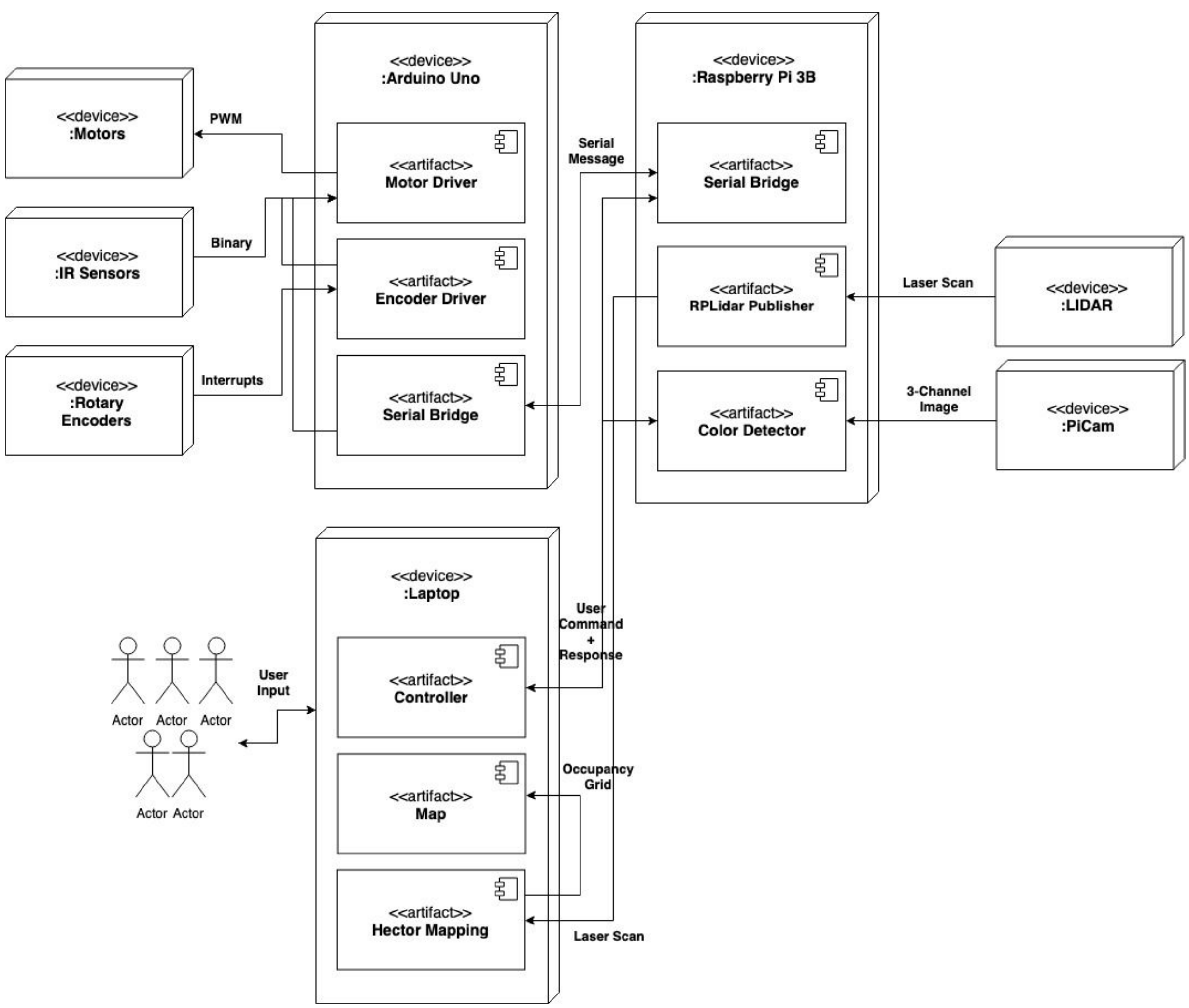
Figure 2. SmokeBot [7]

SmokeBot excels in specific environment scenarios where conventional visual mapping system fails. However, this advantage is also its disadvantage, as, under normal conditions, the specialised system it possesses would not be necessary and could probably be less effective than conventional systems. Additionally, It also requires 15-30 minutes to scan an accident site which is too long in emergency situations[7].

Section 3 System Architecture

Our system architecture is divided into modules installed in three major devices: Arduino, Raspberry Pi, and User.

The modules on the Arduino side manages low-level instructions, such as providing PWM signal for motor movement, handling interrupts from the rotary encoders, and reading the measurement given by the IR sensors. The modules on the Raspberry Pi side, on the other hand, take care the operations related with publishing the ROS messages for LIDAR readings, running the algorithm for colour detection, and relaying information to Arduino side. Lastly, on the user side are the modules dealing with user input, displaying the map + response from Alex, and running Hector SLAM algorithm.



Section 4 Hardware Design

Considerations for Design:

- Weight Distribution and Center of Gravity
    - It is important to ensure that the robot is balanced as this will prevent it from tipping over when making turns or start-stop motions. If such a thing happens, it make cause the robot to become irrecoverable without human assistance.
    - It is important to push the center of gravity low, right over the “axle” of the wheels. This allows for the least wheel slippage, as the wheels are pushed into the ground.
    - During experiments, we found that moving the center of gravity closer to the ball bearing wheel caused the movement of the robot to become unpredictable as the ball bearing frequently jammed, causing the robot to move away from its intended path.
- Motor Load
    - The small motors supplied are not meant to carry heavy loads or to make rapid changes in speed over a prolonged period of time. Hence we designed the robot such that each motor is only required to make small unidirectional movements, reducing the wear and tear of the motor.
- Wire management
    - This was a key consideration in our design as we ensured that as much as possible, wires were under no strain, and were either cable tied or taped together where possible. Jumper wires had their heads taped together to reduce the bending strain on individual male pins. This significantly reduced the rate of breakage of pins. Additionally, we used no breadboards in the design, and instead soldered all connections that were not direct port to port connections. Hence our circuitry was much more consistent and reliable as it reduced the risk of short circuits or breadboard damage.
- Power Consumption
- Peripherals

Section 5 Firmware Design

Arduino side:

1. Calibration (removes the need to manually update hard-coded constants, will not be run during evaluation)
    1. Receive PWM calibration data from Pi through Serial Bridge.
    2. Update the lower and upper bound PWM values for each motor through Motor Driver, such that both motors will move equally fast. The lower and upper bound would correspond to 0% and 100% power respectively, from Pi point of view.
2. Initialisation
    1. Perform handshake with Pi
    2. Begin mapping module in an idle state, i.e map environment in longer intervals, or just take a few initial frames.
3. Sending and Reading of Data
    1. If *'drive'* command is received from Serial Bridge, instruct Motor Driver to drive motor. Mapping module moves from idle to normal operation.
    2. If interrupts connected to the encoders are triggered, send the updated calculated coordinate/ticks, as well as data from any other sensors we are using (data to send will depend on the input of the SLAM algorithm) back via Serial Bridge.
4. Repeat 2

Section 6 Software Design

Raspberry Pi side:

1. Initialization
    1. Perform handshake with both user and Arduino
    2. Shutdown all unused functionalities via power control registers
    3. Position set at (0,0)
2. Run, which allows the modules below run concurrently
    1. Serial Bridge (send and receive data from and to Arduino)
    2. Mapper / Colour Detector (only one at a time)
        1. Encoder data from the Arduino and LIDAR data will be inputs into a Hector SLAM system
        2. Processed data will be streamed to the operator, where it will be visualized
        3. If the operator observes an object, he may choose to initiate colour detection, upon which robot will be stopped, along with the Mapping module
        4. This allows us to choose a few different options to accomplish colour detection, given no expensive tasks are run concurrently
            1. TCS230 colour sensor which will use white LEDs to reflect light off surfaces onto photodiodes, allowing it to measure frequencies of the reflected light (less reliable)
            2. PiCamera module which will use colourspace filtering (HSV or LAB colourspace) to get the mask of the object and output the colour of the object based on the area of the contour obtained (more power-wise / computationally / cost-wise expensive)
    3. Control Interface (this will send data to Motor Bridge and instruct Color Detector to execute + Mapper to stop when want to scan colour)
    4. If no drive command is given, can consider pausing SLAM or idling the mapping process to save power
3. Repeat 2

User side:

1. Initialization (test out TCP/IP connection)
    1. Include ability to verify this through hello handshake

2. <u>Run</u> (the below modules run concurrently)
    1. Controller (receive user input and send to Control Interface)
        1. Directional commands (WASD)
        2. Colour sensing command (c)
        3. Hello (h)
        4. Change motor speed (+/-)
    2. Map (receive the stream of data from Mapper and display on screen)

Section 7 Lessons Learnt and Conclusion

From this project, we learned that we need to plan before we execute the plan. Taking time to plan the process allows us to make effective decisions about how to allocate the resources that are provided by the lab and bought by us. This increases productivity and resources are not wasted. Also, when we faced an issue, we are able to backtrack and identify the issue and this makes debugging much easier.

Additionally, by splitting the workload, we are able to do the project faster and more efficiently. We understand that different people have different ways of tackling a problem, thus having multiple brains working on one component will not only be wasting effort, but it also slows it down as different opinions may disturb the chain of thoughts of the main person that is tackling it. Therefore, we assigned everyone different task according to their capabilities and put everything together at the deadline that we set. Since all modules are interconnected, we have to take information from one another which increases the pressure to complete the task with great accuracy.

However, we made the mistake of purchasing our additional parts too late. We did not expect that the additional parts will arrive so late due to some unforeseen delays in the shipping process. This caused our project to be delayed for a long time due to purchasing parts from online shops. In the future, as we do not have an unreasonably low monetary restriction, we should have sourced for electronic parts locally so that we can start our project much earlier.

Another mistake that we made during the early phases of our project was not judging everyone’s capabilities before assigning tasks. This led to a small delay in work as some of us were not able to catch up with the rest of the group members due to difference in technical skills and knowledge.

References

[1] *API - Colors and Color Matching*. [Online]. Available at: https://picamera.readthedocs.io/en/release-1.12/api_color.html

[2]"State of the art", *En.m.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/State_of_the_art. [Accessed: 10- Apr- 2019].

[3] E. Ackerman and E. Guizzo, “New WALK-MAN Robot Is Slimmer, Quicker, Better at Quenching Your Flames,” IEEE Spectrum: Technology, Engineering, and Science News, 22-Feb-2018. [Online]. Available: https://spectrum.ieee.org/automaton/robotics/humanoids/new-version-of-walkman-is-slimmer-quicker-better-at-quenching-your-flames [Accessed: 17-Mar-2019]

[4] Davis, S. (2018). *The New WALK-MAN: A Look at IIT’s Multi-Faceted Robotic Endeavor*. [online] Power Electronics. Available at: https://www.powerelectronics.com/robotics/new-walk-man-look-iit-s-multi-faceted-robotic-endeavor [Accessed 14 Mar. 2019]

[5] EurekAlert!. (2018). *The humanoid robot WALK-MAN for supporting emergency response teams*. [online] Available at: https://www.eurekalert.org/pub_releases/2018-02/iidt-hrw022118.php [Accessed 14 Mar. 2019]

[6] SmokeBot - Mobile Robots with Novel Environment Sensors for Inspection of Disaster Sites with Low Visibility. [online] Available at https://www.smokebot.eu/project.html [Accessed 13 Mar. 2019]

[7] “SmokeBot – a robot serving rescue units,” Phys.org - News and Articles on Science and Technology. [Online]. Available: https://phys.org/news/2018-06-smokebot-robot.html [Accessed: 17-Mar-2019].