The Implementation of Unmanned Ground Vehicle in Autonomous mechatronics systems

Shusen Zhai

University of South Australia

zhasy061@mymail.unisa.edu.au

Abstract

More and more robots are applied in the human life now, which can be substitute human and replicate human actions. This technology can be used in some dangerous environments, such as manufacture process, military and where humans cannot survive. Robotics technology develops fast in recent years. Thus, students can learn the operation and programming skills about robotics technology in this project. There are six steps to finish the Autonomous Mechatronic Systems project, which are localization, navigation, path planning, mapping, target retrieval and object avoidance. Encoder, Inertial Measurement Unit (IMU) and Vicon system to are applied to localise in localisation. By doing calibration of Fuzzy Logic controller and tuning the PID controller are important processes in navigation. A star algorithm, which is widely used in pathfinding and graph traversal, is chosen in path planning. For mapping, grid mapping is used in this project. By using OpenCV libraries, thresholding, opening and closing functions can be used to separate each target colour in target retrieval. Infrared Sensor and Ultrasonic Sensor are used to avoid obstacles. Results of this project include the encoders and Vicon system readings for calibration, the map of the maze, the shortest moving path, different colour range of the three targets and pick up requirements. Analysing these results is useful to improve the performance of the Unmanned Ground Vehicle (UGV) in the future.

Keywords

Autonomous Mechatronic Systems, Fuzzy Logic Controller, HSV, IMU, Infrared Sensor, OpenCV, PID Controller, Python, Unmanned Ground Vehicle, Ultrasonic Sensor, Vicon.

1. INTRODUCTION

Robotics technology, which is a confluence science including mechanical engineering, sensor fabrication, manufacturing techniques and advanced algorithms, plays an important role in the real life and scientific fields. The application of robots has promoted the development of robotics and other sciences (Metta et al. 2001). Thus, students can learn skills and knowledge from different disciplines in the study and practical of robotics. To finish this project, students need to use relevant programming knowledge and algorithms.

In robotics, building a map from the environment is a vital task. It has a close connection with other parts, such as localization, path planning and obstacle avoidance (Romero & Cazorla 2012). The construction of a map when a robot's locations are known is comparatively easy. To construct a map, roads and walls should be added to represent the static elements first, which can allow the robot to do path planning for long distances or long period. Then, to make the robot get instant information like dynamic obstacles, the map need take the last sensor readings which is a concept of instant representation to make robot react (Matta-Gómez, Del Cerro & Barrientos 2014).

To navigate the robot, path planning is a vital problem need to solve, which is focused on how to move a robot from one position to another position. Generally, path planning is concentrated on creating algorithms which can make suitable actions by using some complex geometric models (Duchoň et al. 2014). It can be divided into two sorts: global path planning with all information of the identified environment in the robot and local path planning in partially or fully unidentified environment.

Path planning Algorithms need a complete information about the environment and should be capable to produce a new path when the environment. A* algorithm is a widely-used path planning algorithm, which can be used in metric and topological configuration space by combining heuristic searching and the shortest track searching changes (Ismail, Sheta & Al-Weshah 2008). The formula for A* algorithm can be expressed as f(n)=g(n)+h(n). In this formula, g(n) is the length of the track from start point to the goal point through the certain series of cells, and h(n) is heuristic distance of the cell to the goal. Compared with other path planning algorithms, such as Dijkstra algorithm, A star algorithm can save more time and is more suitable to use in situation where to find the path quickly is essential (Duchoň et al. 2014).

Image processing is very important for target retrieval. OpenCV is an open source library aiming at providing tools to solve computer vision problems, which contains logical and arithmetic operations, such as face detection, feature matching and tracking (Pulli et al. 2012). OpenCV library includes several modules which are easily to understand. Generally, all functions are concentrated on computer vision problems, and they are described within the name space cv (Culjak et al. 2012). In addition, the image processing adopts HSV colour model, which is another expression of RGB colour space. HSV colour model uses three basic features to define colour, which is hue, saturation and luminance separately. Hue is the basic feature of colour, which ranges from 0 to 360. Saturation shows how higher and purer of the colour is, which ranges between 0 and 100%. Luminance shows brightness that ranges from 0 to 100%. Using RGB cannot separate colour information from luminance, but HSV model can be used to separate image luminance from colour information. (Shuhua & Gaizhi 2010). Thus, compared to RGB, HSV colour model will be much better for colour texture analysis (Chernov, Alander & Bochko 2015).

Thresholding is an important part to abstract target image. Choosing the right threshold is a necessary task for target edge detection and some advanced edge detectors also use two thresholds. It is the process of creating binary image based on the threshold value. The pixel will be white if the value greater than threshold value. Otherwise, a pixel will be black (Kim & Jun 2011). For example, the canny operator, which is one of the gradient operators, can produce the low and high edge maps separately and abstract points from low map into high threshold edge map (Culjak et al. 2012). Different targets can be used different thresholding parameters to identify. Also, histogram thresholding, which makes image to binary image, can use different values to separate the target and background (Tsai & Liu, 2015).

Opening and closing processing are the next step after thresholding. They are both derived from erosion and dilation, which are two important operators from mathematical morphology (Haralick & Shapiro 1992). Opening function include an erosion followed by a dilation with a same structuring element. Thus, opening can remove noise outside targets. On the contrary, closing function contain a dilation followed by an erosion with a same structuring element. Similarly, closing can removes small holes in the foreground, changing small island background into foreground (bright) (Toet 1989).

The contour detection is also an important part of image processing, which can help UGV to classify the positon and size of targets. Contour is composed of boundary pixels of the target, and contour detection is to store boundaries of targets in the image (Gurav & Kadbe 2015). The contour of target can be stored in a tree structure with discerned internal and external contour. In OpenCV, the cv2.findContours function is used to store contours and cv2.contoursArea is implemented to calculate area of targets (Kim & Jun 2011). Thus, the UGV can get the position of the target, and then finish target retrieval.

2. Methodology

2.1 Project Scope

This project includes several parts, which is localization, navigation, mapping, path planning and target retrieval.

2.1.1 Physical Block Diagram

Figure 1 is the physical block diagram. This diagram shows each physical part and the structure of the UGV that will be used to applied in this project.

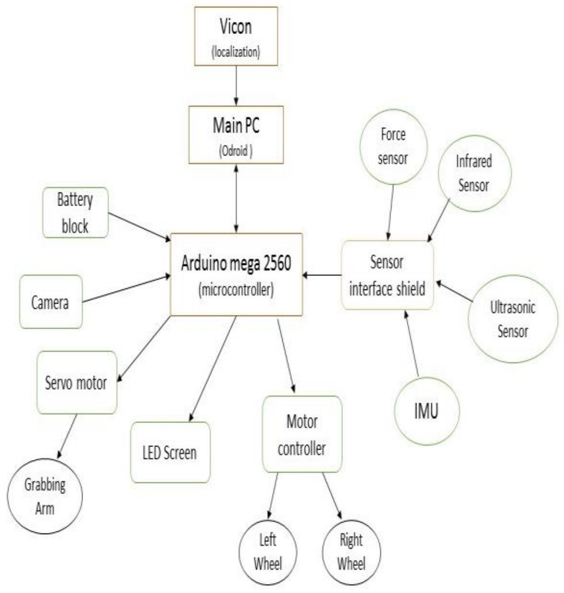


Figure 1. The Physical Block Diagram

2.1.2 Functional Block Diagram:

Figure 2 is the functional block diagram. This diagram indicates the flow of whole process to do this project.

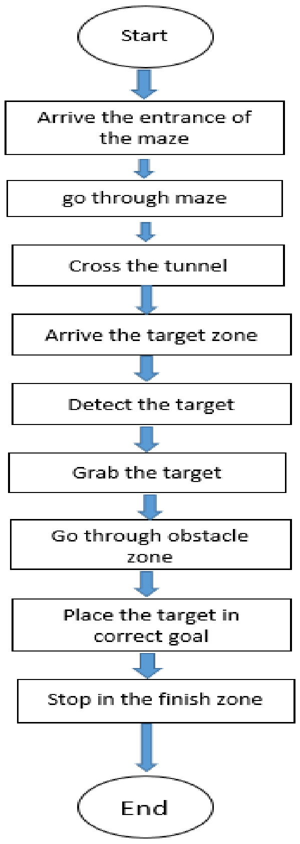


Figure 2 The functional block diagram

2.2 Localisation

Multiple sensors were introduced in this project for obtaining the pose value of the UGV. Vicon system is an external localization sensor. It has very high accuracy comparing to other internal sensors. Encoder and IMU were introduced in this project as internal sensors. They were less accuracy than the Vicon sensor. Ideally, Vicon can provide correct pose value through this project. However, the path of the project was design with a Vicon signal blocked tunnel. There was also 10Hz random noise with the Vicon signal outside the tunnel. Therefore, our localization used both three sensors. Inside the tunnel, the pose value was provided by the fusion of IMU and encoder. Outside the tunnel, the fusion with both three sensors was used. In the python script, 'if' statement was used to switch the signals inside and outside the tunnel.

2.2.1 Encoder calibration

This calibration is to make the encoder pose values closed to the Vicon pose values as much as possible. The encoder was calibrated during the practical study. However, extra calibration was also necessary. The calibration was focused on calibrating linear displacement and angular displacement of the UGV. In the calibration, ten trials of data were taken for each linear (x and y poses) and angular (heading value calculated by x and y value) displacement provided by encoder and Vicon. By using the equation of calculating confidence intervals $\mathrm{D} = \mathrm{mean} \pm \mathrm{CV}\sigma/\sqrt{n}$, where CV is the critical value, σ is the standard deviation and n is the number of samples. The idea outcome is the confidential interval crossing zero. Otherwise, we need to calculate the percentage of difference between encoder and Vicon and modify the value of w_base or e_res in the coding until the confidence intervals meet the requirement.

2.2.2 Sensor Fusion

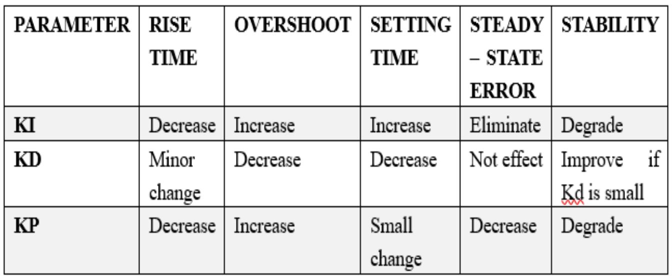
For obtaining more accurate pose value, we decided to fuse the sensors. when the UGV was outside the tunnel, there is the Vicon signal with random noise. So, we fused Vicon, encoder and IMU sensors together to overcome this problem. When the UGV was inside the tunnel, we fused the encoder and IMU for the pose value. We determined the initial weight by calculating the variances between the sensors pose value. Then ran the UGV with the initial weight in the lab and saw the performance. Modify the weight with it performance.

2.3 Navigation

Fuzzy proportional control was used in navigation control. The control method is based on what we learned in the practical.

2.3.1 PID controller

The reason to use a PID controller is that the it can operate the mechbot's two wheels independently. By using the PID controller, the mechbot's wheel speed can also be controlled by the inserted set point. There are three PID gains that are set to make the motor output speed reaching the set point, integral(Ki), derivative(Kd) and proportional gain(Kp). This PID controller also tell the mechbot to stop when its integral error is zero. the table below show how to calibrate this PID controller by changing three gain values.



PARAMETER	RISE TIME	OVERSHOOT	SETTING TIME	STEADY – STATE ERROR	STABILITY
KI	Decrease	Increase	Increase	Eliminate	Degrade
KD	Minor change	Decrease	Decrease	Not effect	Improve if Kd is small
KP	Decrease	Increase	Small change	Decrease	Degrade

Figure 3

2.3.2 Fuzzy logic and proportional control

The knowledge used in this section was based on the practical. In fuzzy logic, the goal location is set in coding. By using the localization system, the UGV can know where is the direction to forward. The angular error and distance error is also calculated by the method in the figure below. By setting the values of angular error and distance error in coding, the UGV can correct its heading error once the error is beyond the setting and stop the UGV within the maximum distance error.

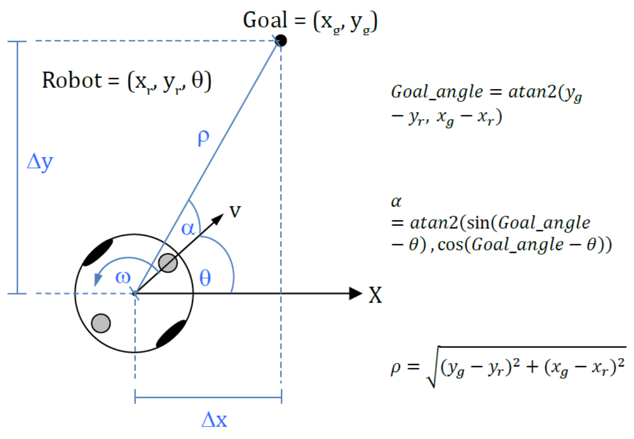

Goal = (x_g, y_g)
Robot = (x_r, y_r, θ)
Δy
ρ
α
v
ω
θ
X
Δx
$Goal\_angle = atan2(y_g - y_r, x_g - x_r)$
$\alpha = atan2(\sin(Goal\_angle - \theta), \cos(Goal\_angle - \theta))$
$\rho = \sqrt{(y_g - y_r)^2 + (x_g - x_r)^2}$

Figure 4

In fuzzy proportional control, based on the method of fuzzy logic, there are three control P values added in. The gain values are used for controlling the action of UGV to go forward, rotation and small angular correction. With the basic setting in fuzzy logic, the UGV can operate smoothly by calibrating three control P value.

2.4 Mapping

For mapping, the 'Tiled' program can help to create a map. The grid map was used as the map in this project, which is clear and easy to do the following path planning. First, the original maze map need to be reduced 250 times, the length and width from 3500*3000 millimetre (mm) to 14*12 mm. Thus, each grid size is 250*250 mm. Next, using different colour to represent walls, freeways, start zone and end location. Using '0' to show the free access in the map that the UGV cannot go through, and using '1' to show the walls which the UGV cannot pass. In addition, the diameter of the UGV was about 240 mm, so each grid size is big enough for the UGV to go through. Although these walls in the grid map occupy a part of space of the free access, it doesn't affect the path planning and the running of the UGV. Also, because the free access in the grid map is smaller and the wall is bigger than the real maze, the chance of the UGV to hit walls can be reduced.

2.5 Path Planning

A* algorithm combines the advantages of the Best-First Search and the Dijkstra algorithm, which is an effective and direct method to obtain the shortest path. The heuristic search can improve the efficiency of the algorithm, and it can guarantee the finding of an optimal path which is based on the evaluation function. In the formula f(n)=g(n)+h(n), g(n) represents the actual distance from the starting point to the arbitrary vertex n, h(n) represents the estimated distance from the arbitrary vertex n to the target vertex (depending on the difference in the evaluation function adopted).

2.6 Target Retrieval

This subsystem can be divided into three parts, which are colour detection, picking up and dropping off targets.

2.6.1 Colour Detection

To do the colour detection, the camera in the front of the robot was applied to capture the image, and OpenCV library was used to do the image processing. First, the original image format was converted from RGB into HSV. This reason for using HSV model is that it can directly express colour light, shade, tone and brightness. In RGB model, colour information cannot be separated from luminance, which is not suitable for this project. Next step is image thresholding, the HSV color range of each target (Blue, Green and Red) can be captured from three image value windows, which are Hue, Saturation and Value. After getting these ranges of each colour, adjusting and testing these range can get the threshold image. After thresholding, the image still has some noise, and the use of opening and closing function in OpenCV can help to remove the noise. Opening is doing erosion and then dilation, so it can remove noise from the background. Closing is doing dilation and then erosion, so it can change small island background into foreground. Thus, using both opening and closing function can remove outside and inside noise of target separately. Next step is to get the contour of target. Cv2.findcontours function was used to find the contour of target. Then using 'cv.contourArea ' to calculate the area of target and using 'cv.boundingRect' to get the range value of contour which includes the top left corner coordinate(x, y), width(w) and height(h) value of the target in the image window.

2.6.2 Picking up and dropping off targets

According to the values derived from above part, which are x coordinate, w and h value, the distance between the robot and targets can be defined. Similarly, the target position for target in left or right of the robot centre can use the difference between target centre and image centre to define. The target centre value can use x value plus half of target's width to represent (x+w/2), and the image centre value is half of the image width, the following formula shows the principle to define target position.

$$\text{Position} = \begin{cases} left \quad x + \frac{w}{2} - \frac{image\ width}{2} < 0 \\ middle \quad x + \frac{w}{2} - \frac{image\ width}{2} = 0 \\ right \quad x + \frac{w}{2} - \frac{image\ width}{2} > 0 \end{cases}$$

Thus, when the value of the equation x+w/2-216 is greater than 0, the target is in the right of the robot centre, and when this value smaller than 0, the target is in the left. Also, when the target is in the gripper, this absolute value is less than a value. Next according these above values, the order of picking up and drop off can be set up. When the target is far from the robot, the robot will go forward and adjust heading to make the position value less than in gripper value. Finally, the robot will stop until the target the width, height and position value reach the in-gripper value, and then the gripper can carry out gripper open, close, arm up and arm down function by using servo command to achieve picking up and dropping off.

2.7 Obstacle Avoidance

There are the ideas and methods to do the obstacle avoidance.

2.7.1 Ideas of realizing obstacle avoidance

To avoid the obstacle, detecting the existence of the obstacle is the first and most important task in this part of the project. To approach this, sensors are needed to be framed on the UGV.

The sensor which used for detecting the obstacle in the project is infrared sensor after the research and comparing. Infrared sensors work by sending infrared light of a certain frequency, and then detect if some light has been reflected back to the sensor. The most common is a digital output that indicates if an object is detected. Many of them can choose to enable or disable.

Three infrared sensors are installed on the UGV like the picture and it should be enough to detect if there are some obstacles in front of the UGV. The angle between the adjacent sensors is 45°.

When the sensor detects the obstacle, the UGV will start spinning clockwise 45° and detect again, and it keeps doing this until all 3 sensors detecting nothing, then the UGV will go straight and follow the route of the mapping.

2.7.2 Methods of realizing the idea

To achieve the idea, correctly programming of the system is necessary.

Because the UGV is spinning to find a direction which has no obstacle in front of it, a for loop code should be the main loop, and put the if statement in the loop to check whether it fits the condition, then sending the true condition tell the UGV when it's good to run.

2.8 Integration

For the integration, first step is to publish and subscribe to data on ROS network. The pose and image processing output data can use 'rospy.Publishser' to publish, and in the integration script, these outside data can use 'rospy.Subscriber' to subscribe. Next step is to define the name of the data from Subscriber in the integration script. Then defining each function and publish some data, such as motor and servo data. Finally, setting orders for these functions and put them in the main loop, and set enough sleep time between some functions to ensure robot run each part accurately.

3. Results

3.1 Localisation

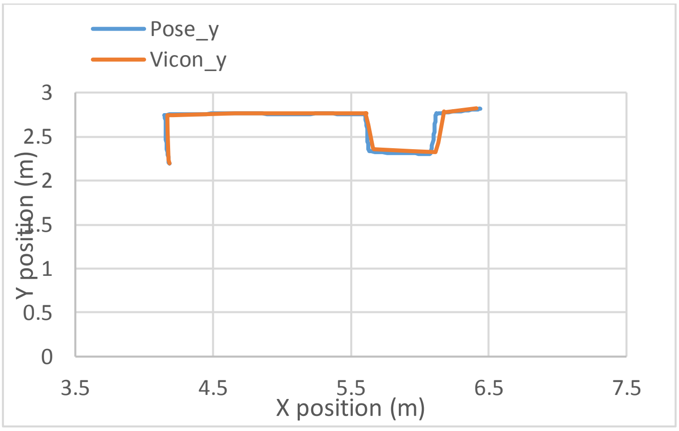


Figure 5.

This figure shows the route estimation of robot from start point to target zone before calibration. There is no Vicon signal in the tunnel. The difference between encoder and Vicon still need improve to avoid hit the wall.

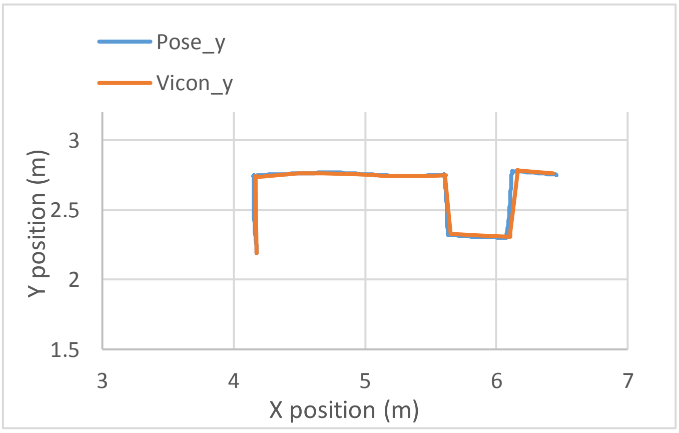


Figure 6

This figure shows the route estimation of robot from start point to target zone after adjusting w_base value, the difference becomes small and it can be acceptable.

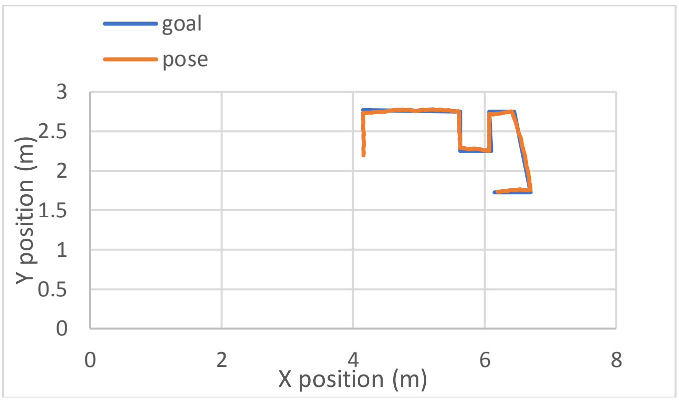


Figure 7

3.2 Mapping

3.2.1 Map conversion

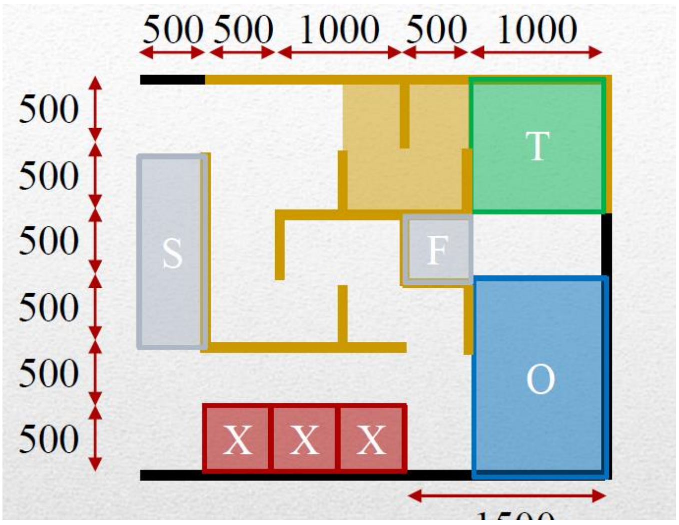


Figure 8. The original maze

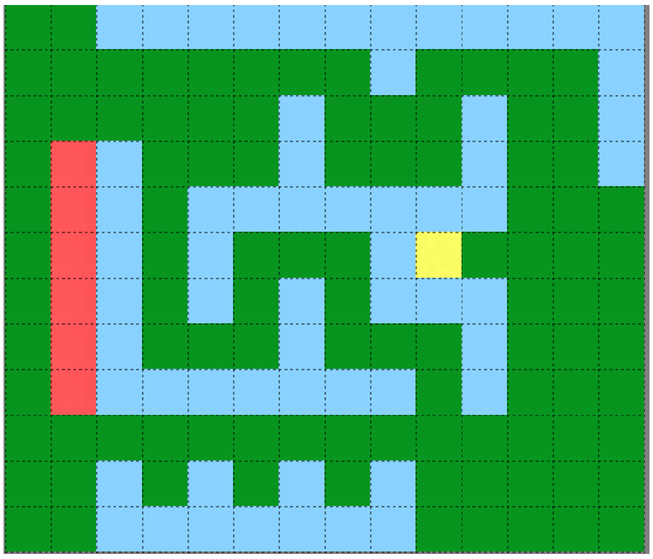
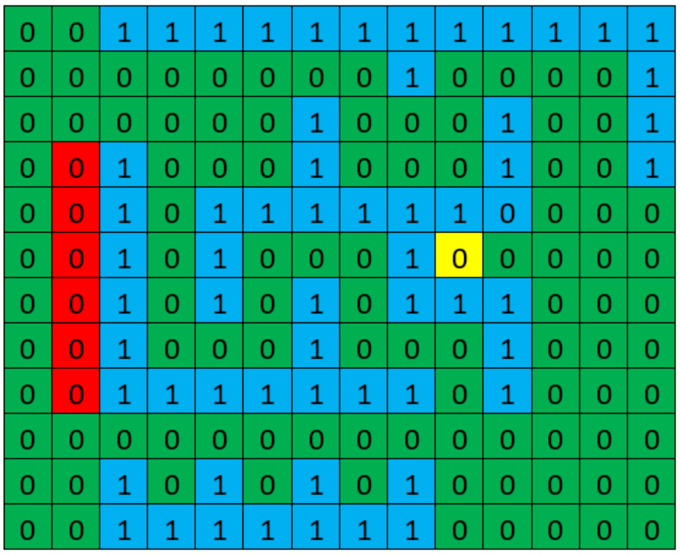


Figure 9. The grid map

Figure 8 is the original maze. Figure 9 is the grid map which was used the 'Tiled' program to build. In this grid map, the original size was reduced 250 times and each grid size is 250*250mm, this map size is 14*12. Red grids represent start zone, blue grids represent walls, yellow grid represents end position and green grids represents free access. In the following figure 10, the free access use 0 to represent, which the UGV can go through, and walls use 1 to1 represent that the UGV cannot pass.



0	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	1	0	0	0	1	0	0	1
0	0	1	0	0	0	1	0	0	0	1	0	0	1
0	0	1	0	1	1	1	1	1	1	0	0	0	0
0	0	1	0	1	0	0	0	1	0	0	0	0	0
0	0	1	0	1	0	1	0	1	1	1	0	0	0
0	0	1	0	0	0	1	0	0	0	1	0	0	0
0	0	1	1	1	1	1	1	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0	1	0	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0	0

Figure 10

3.3 Path Planning

3.3.1 Setting Path

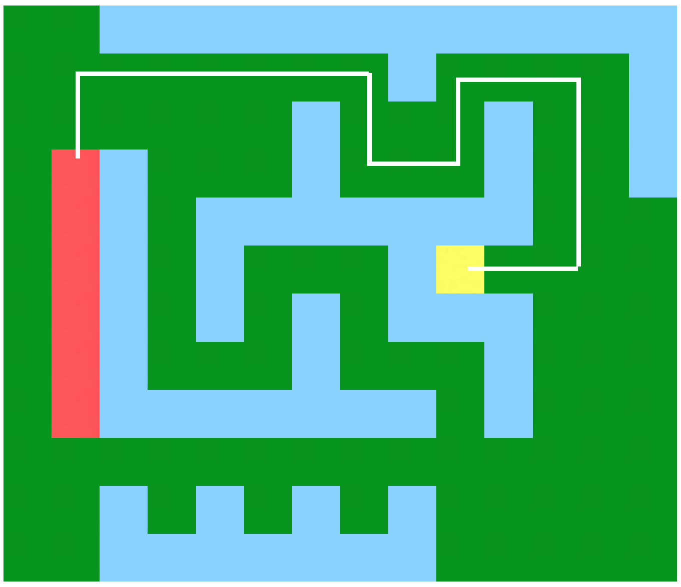


Figure 11

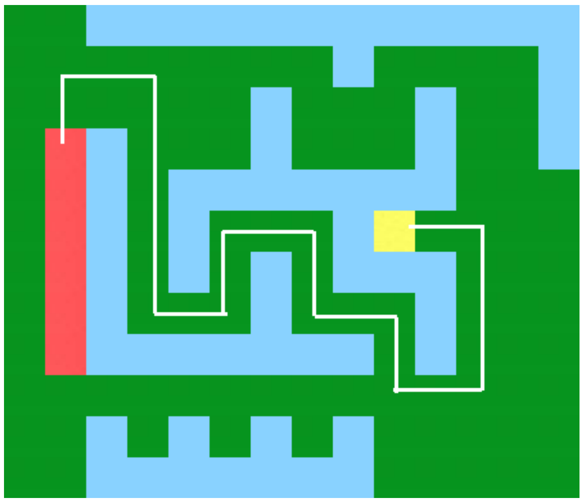


Figure 12

For path planning, A* algorithm was used to find the shortest path from the start zone to the end location. Figure 11 and figure 12 are two path for this project, and the white line represents the path that UGV go through. Figure 11 is the path that go through the tunnel to the end location, and figure 12 is the path that go without tunnel but go through the obstacle zone to the end location. Because go to tunnel has 5 marks in the demonstration, and the UGV can go through the tunnel without hitting the wall, our group used the path in figure 11 as our project path.

3.4 Target Retrieval

3.4.1 Colour Detection

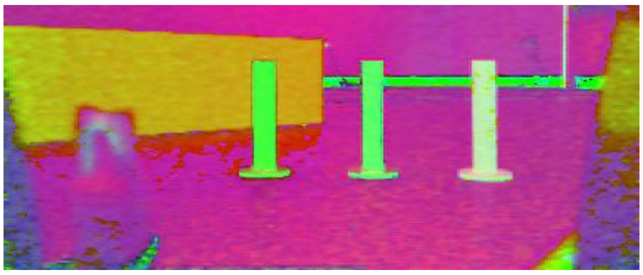


Figure 13 HSV image

Figure 13 is the HSV image of the three targets converted from RGB image. The blue, red and green HSV colour value attained from the HSV image are shown in the following table.

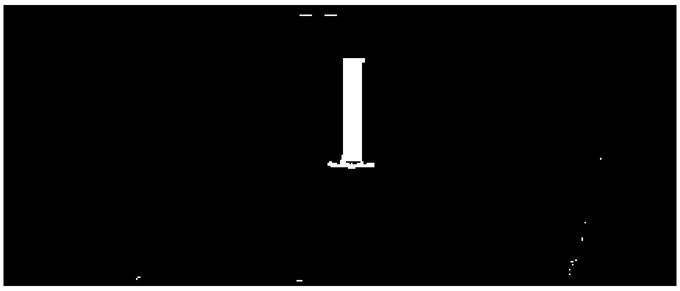


Figure 14 Thresholding image

The figure 14 shows the target image after thresholding. There is some noise outside and inside of the target, so the image need use the opening and closing function to remove noise.

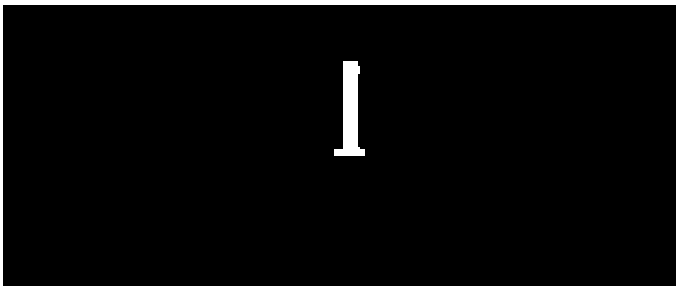


Figure 15

Figure 15 is the target image after using opening and closing function from OpenCV. The outside and inside noise has been removed successfully. The following step is to define the contour of target.

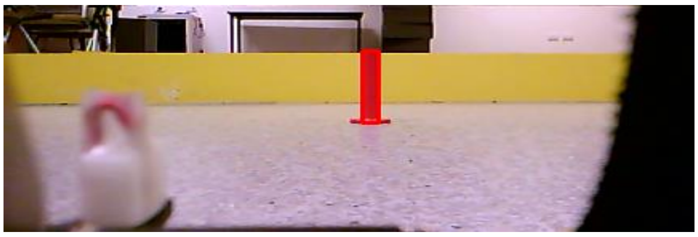


Figure 16

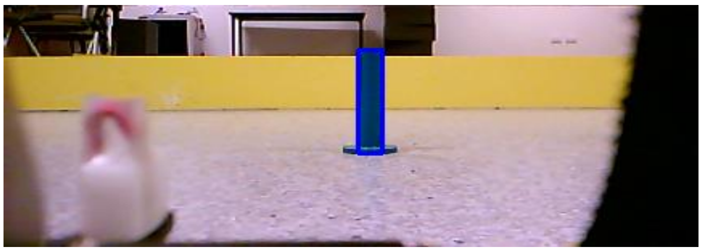


Figure 17

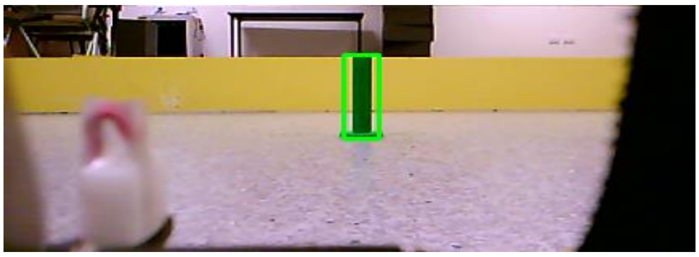


Figure 18

Figure 16, 17 and 18 shows the contour of three targets which were used the cv. Rectangle function to draw after defining contour ranges. In target zone, the furthest distance between targets and the UGV is 1.272m, and the UGV can accurately detect the target in this range.

3.5 Obstacle Avoidance

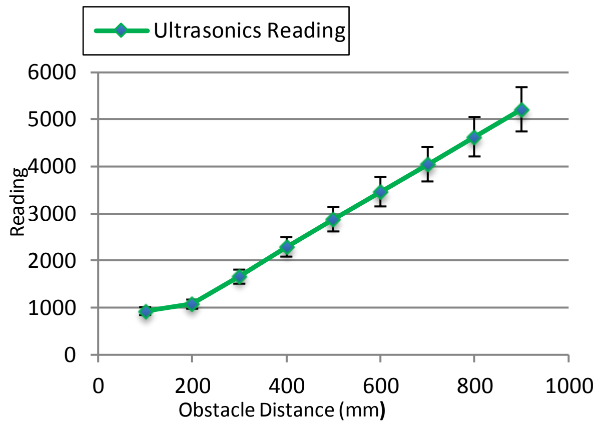


Figure 19

Figure 19 is the reading of ultrasonic sensor, and the reading increases with the obstacle distance.

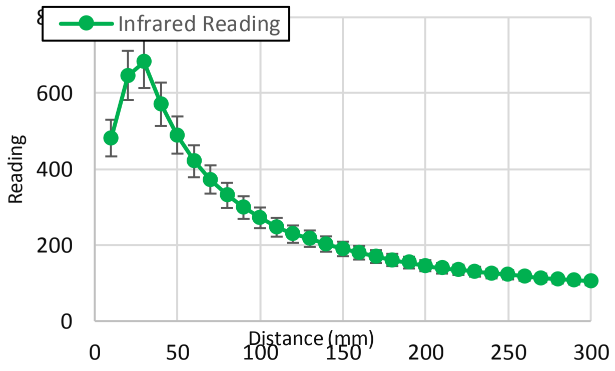


Figure 20

Figure 20 is the reading of Infrared sensor. When the distance greater than 50mm, the reading will decrease with the increase of distance.

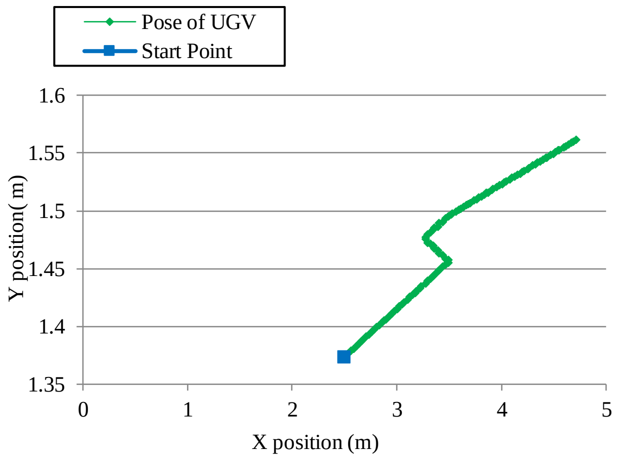


Figure 21

Figure 21 is the test for the obstacle avoidance. The UGV can go through the obstacle and reach the set point. However, sometimes the UGV will touch the wall, but the frequency is not high, about one time in ten times test.

3.6 Final performance

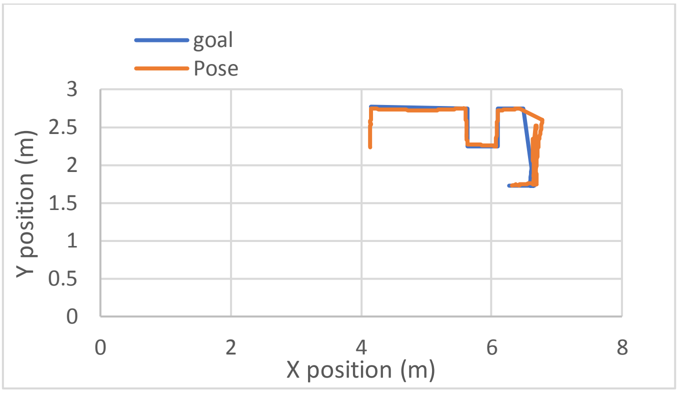


Figure 22

In this project, because Vicon has random noise, the final navigation and localisation combined encoder, IMU and Vicon together. The fusion for the encoder, IMU and Vicon is 0.2, 0.1, 0.7 separately, and in the tunnel without Vicon signal, the fusion for encoder and IMU is 0.3 and 0.7. The motor max speed is 100 for going forward and 60 for turning. Figure 22 shows the path for the demonstration, which includes target retrieval and finish in the end location. Because the group final plan not to go the obstacle zone, for the demonstration, the UGV will pick up and drop off all targets in the end location and finish in the end location.

4. Conclusion

This project includes the use of Python and OpenCV library. Mapping and path planning are prerequisites for localisation, navigation and obstacle avoidance. Mapping needs create a suitable map and path planning is required to find the shortest path from start zone to end location. This two subsystem need student to review relevant literature, and to do repeated test after setting up. A good localisation can provide the robot with accurate positions for navigation, which need combine the encoder, IMU and Vicon system together. Using the fuzzy proportional control can make the UGV have a stable output in navigation. Using OpenCV to do image processing can provide information for target retrieval. To pick up and drop off the targets, some motor commands and enough sleep time are required to perform this section well. In conclusion, AMS project is a comprehensive project, which can show the use and future development of unmanned ground vehicle.

References

[1] Chernov, V, Alander, J & Bochko, V 2015, ‘Integer-based accurate conversion between RGB and HSV color spaces’, Computers & Electrical Engineering, vol. 46, no.2, pp. 328-337.

[2] Culjak, I, Abram, D, Pribanic, T, Dzapo, H & Cifrek, M 2012, ‘A brief introduction to OpenCV’, in MIPRO(ed.) 2012 proceedings of the 35th international convention, pp. 1725-1730.

[3] Duchoň, F, Babinec, A, Kajan, M, Beňo, P, Florek, M, Fico, T & Jurišica, L 2014, 'Path Planning with Modified a Star Algorithm for a Mobile Robot', Procedia Engineering, vol. 96, pp. 59-69.

[4] Gurav, RM & Kadbe, PK 2015, ‘Real time finger tracking and contour detection for gesture recognition using OpenCV’, Industrial Instrumentation and Control, pp. 974-977.

[5] Haralick, RM & Shapiro, LG 1992, Computer and Robot Vision, Addison-Wesley Publishing Company, vol. 1, no. 5, pp. 174 - 185.

[6] Ismail, AT, Sheta, A & Al-Weshah, M 2008, ‘A mobile robot path planning using genetic algorithm in static environment’, Journal of Computer Science, vol. 4, no. 4, pp. 341-344.

[7] Kim, J & Jun, H 2011, ‘Implementation of image processing and augmented reality programs for smart mobile device,’ In Strategic Technology (IFOST), 2011 6th International Forum on, vol. 2, pp. 1070-1073.

[8] Matta-Gómez, A, Del Cerro, J & Barrientos, A 2014, 'Multi-robot data mapping simulation by using microsoft robotics developer studio', Simulation Modelling Practice and Theory, vol. 49, pp. 305-319.
[9] Metta, G, Sandini, G, Natale, L, & Panerai, F 2001, ‘Development and robotics’, In Proceedings of the IEEE-RAS International Conference on Humanoid Robots, pp. 33-42.
[10] Pulli, K, Baksheev, A, Kornyakov, K & Eruhimov, V 2012, Realtime computer vision with OpenCV, Queue, vol. 10, no. 4, p. 40.
[11] Romero, A, & Cazorla, M 2012 Topological visual mapping in robotics, Cognitive processing, vol. 13, no. 1, pp. 305-308.
[12] Shuhua, L, & Gaizhi, G 2010, ‘The application of improved HSV color space model in image processing’, In Future Computer and Communication, vol. 2, pp. 2-10.
[13] Thrun, S & Bücken, A 1996, ‘Integrating grid-based and topological maps for mobile robot navigation’, In Proceedings of the National Conference on Artificial Intelligence, pp. 944-951.
[14] Toet, A 1989 A morphological pyramidal image decomposition, Pattern recognition letters, vol. 9, no. 4, pp. 255-261.
[15] Tsai, CY & Liu, TY 2015, ‘Real-time automatic multilevel color video thresholding using a novel class-variance criterion’, Machine Vision and Applications, vol.26, no. 2-3, pp. 233-249.