Cost-effective, and Algorithmic Method for Autonomous Terrestrial Robot Localization, and Map-building using Multi-ultrasonic Sensors

Jaskirat Singh, Neel Adwani, and Ashish Karn

***Abstract*— The importance of Autonomous mobile robots is increasing for individuals and industries, enriching the automation and robotics sector. Various algorithmic strategies have been incorporated into mobile robotic technology, allowing experts to use it while also making it famous among novices. However, implementing such algorithms on a robot to perform automated tasks can be expensive and complex, as multiple sensors will be required for gathering data, and the high price of multisensory systems. This paper provides an algorithmic, and cost-effective solution by developing the optimized algorithm, using multi-ultrasonic sensors compared to other expensive sensors, implemented within the terrestrial robot. The proposed approach is based on four fundamental processes: sensing, localization, map generation, and humanoid visualization. Three ultrasonic sensors (HC-SR04) are positioned in three distinct axes on our terrestrial robot, collecting data points from obstacles in three different directions within the linear based arena. The acquired data points aid the terrestrial robot in creating its local environment map in this new method. The terrestrial robot navigates confidently via the achieved linear path using a variety of range data. The data set is then utilized to visualize the humanoid mapped environment. Experimental results have shown that our optimized algorithm is quite efficient with way less-expensive multi-ultrasonic sensors, making it a very cost-effective hardware solution for real-world automation problems.**

I. Introduction

Autonomous mobile robots have been playing a vital role in the field of automation and robotics by having a good amount of increased efficiency and productivity. Navigation and mapping have become very active research in the past. A burdening issue strikes during the interaction between sensing and navigation planning when the data is inaccurate. The applications range from outer space exploration to mission planning in extreme environments. In all these situations, to achieve a reasonable level of autonomy, this paper represents the three basic requirements are sensing, analyzing, and algorithmic reasoning which helps in endowing the system with better autonomy. The task of sensing is achieveded by the onboard sensors that gather information about the robot and environment. In analysing, the environment is analysed and canvased as a map for the robot. [6], and [7] explains the research conducted in unknown environments that led to the navigation and control of robots with sensorial devices but they generate lots of noisy data which leads to less optimization solution compared to our research. Algorithmic reasoning is accomplished by devising algorithms that make use of the obtained information to generate commands for the robot to execute the given task.

To make the use of autonomous robots more feasible in real-life applications [23] that could span the entire autonomic system, it is necessary to understand the trade-off between costs and benefits which has been discussed in the recent studies of "A Trade-off Analysis of a Cloud-Based Robot Navigation Assistant Using Stereo Image Processing," by J. Salmerón-Garcı´a, P. Íñigo-Blasco, F. Dı´az-del-Rı´o and D. Cagigas-Muñiz,

The use of expensive sensors (e.g., LiDAR [16], Microsoft Kinect, Intel Real sense, Radar, Stereo, or Mono Cameras) is prevented in favor of cheaper sensing devices (e.g., Ultrasonic Sensors). Creating a low-cost autonomous vehicle by R.W. Wall, J. Bennett, and G. Eis [9] states the study of accomplishing navigation using a microcontroller, GPS, and ultrasonic sensors, mounted on a modified commercial model radio control car chassis but this research increases the interconnectivity of the sensors failure. [8], [10], and [11] have developed various controlling methods using infrared, ultrasonic, and various robust sensors, which demonstrated the various results in navigation and mapping.

Simulations and experimental studies have taken into account ultrasonic sensor usage due to its good range and work found in Nair et al [2] talks about grid mapping after collecting information from the ultrasonic sensor. The sensor being noisy generates unnecessary data points, regarded as noise, that results in outliers in the map due to specular reflections. To prevent the issue of outliers [17], image processing techniques are used. Some other recent works in this field include the Simultaneous localization and mapping [18] of a wheel-based autonomous vehicle with ultrasonic sensors by Jung et al. [1], Multi-ultrasonic sensor fusion for mobile robots by Zou Yi; Ho Yeong Khing; Chua Chin Seng; Zhou Xiao Wei [5], and the core concept of this study can be found in the Evaluation of ultrasonic sensors in robot mapping by Nair et al [2]. Furthermore, mapping and navigation for mobile robots using ultrasonic data have previously been presented by Kurz [3]. The "Kidnapping" Problem, is a major problem in robotics that

Jaskirat Singh is with the Informatics Cluster, School of Computer Science, University of Petroleum and Energy, Dehradun, Uttarakhand-248007(email – juskirat2000@gmail.com)

Neel Adwani is with the Informatics Cluster, School of Computer Science, University of Petroleum and Energy, Dehradun, Uttarakhand-248007(email – neeltr.n@gmail.com)

Dr. Ashish Karn is with the Mechanical Cluster, School of Engineering, University of Petroleum and Energy, Dehradun, Uttarakhand-248007(email – akarn@ddn.upes.ac.in)

has been solved in [1]. To enhance robot navigation in interior situations, low-cost ultrasonic sensors, incremental encoders, and grid-based probabilistic models are employed by Anousaki and Kyriakopoulos [4]. The range data from ultrasonic sensors are continually sampled for model development, and a map is produced and updated while the robot travels within the workplace. The occupancy grid is the foundation of the local world model. The world model derived from the range data is based on online segments as a geometric primitive. Methods like the Hough transform and clustering are used to extract these characteristics. The perceived local environment model is merged with ultrasonic sensor data in a localization strategy to estimate the current location and orientation of the mobile robot, which is done using an extended Kalman filter.

The work done by S. Kodagoda in [12] regarded a probability-based solution for building maps, which uses the information about time of flight from an ultrasonic sensor. The researchers considered discrete ultrasonic observations at continuous time intervals, by revolving the attached shaft in minor angles, which are incrementally merged into partial planes to produce a realistic representation of an environment that is amenable to sonar [19] localization. The proximity readings were averaged to suppress the transient errors and results were further refined with the help of the sonar probability model that lead to the creation of the occupancy grid.

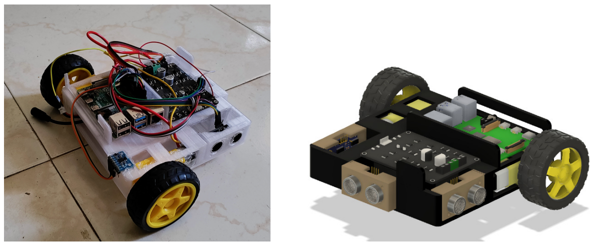

Fig. 1. Terrestrial Robot ran in real environment(Left), and CAD model(right)

However, this paper has implemented the use of the ultrasonic sensor for localization, and mapping giving the real-time environment for adapting and path to the terrestrial robot to navigate through the linear motionary trajectories using our optimsed algorithm, giving more proportionate datapoints for map. Fig. 1, above shows the real-time testing of the terrestrial robot.

II. METHODOLOGY

A. 3-D Architecture

The prototype of this terrestrial robot was designed using Autodesk Fusion 360 and the chassis was printed using Creality Ender 3 with PLA filament, consisting of Raspberry Pi 4 Model B as the microprocessor, three HC-SR04 Ultrasonic Sensors for gathering data points, Cytron MDD10A Motor Driver for supplying voltage to the motors, and two battery-operated motors with wheels, and a castor wheel. The three-view of this terrestrial robot has been presented in fig. 2.

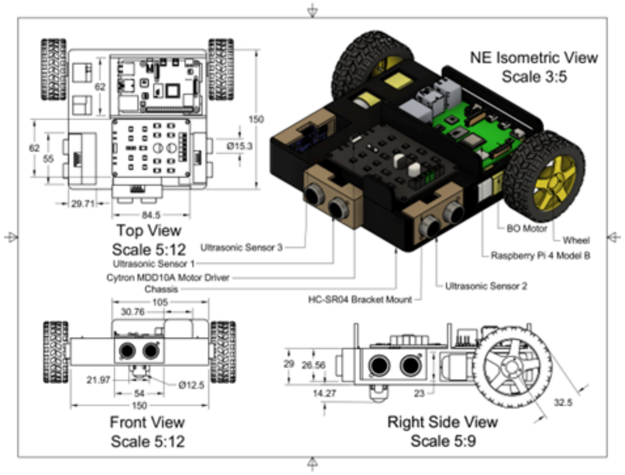

Fig. 2. Conceptual 3D Model Design

B. Hardware Interfacing

To design an algorithm that will be integrated into a terrestrial robot or hardware, it is unquestionably necessary to first construct a hardware circuit. Hence circuit used for the placement and mounting hardware on the robot is described in this section.

Hardware Interfacing is achieved by assembling the circuit on the robot. The circuit comprises various components which are explained in *section II-A*.

As Ultrasonic Sensor 1 has four different pins namely Vcc, Trig (output), Echo (input), and GND where Vcc is connected to Raspberry Pi's pin 4(5V PWR), Trig connected to pin 7(GPIO 4), Echo connected to Pin 11(GPIO 17). Ultrasonic Sensor 2 Vcc gets connected to Vcc of Ultrasonic Sensor 1, Trig gets connected to RPi's pin 37(GPIO 26), Echo gets connected to RPi's pin 35(GPIO 19), and GND gets connected to to RPi's pin 9(GND). Similarly, Ultrasonic Sensor 3's Vcc, Trig, Echo, and GND get connected to RPis' pin 2(5V PWR), pin 29(GPIO 5), pin 31(GPIO 6), and pin 6(GND) respectively. The circuit diagram has been shown in fig. 3.

The Gyro Sensor (MPU-6050) has four different pins namely VCC, GND, SCL, and SDA where VCC and GND are connected to Vcc and GND of Ultrasonic Sensor 3 which is further connected to pin 2(5V PWR) and pin 6(GND) respectively.

The circuit has an MDD10A motor driver attached to surf the motors. The right side of the motor driver comprises five different pins namely DIR1, PWM1, DIR2, PWM2, and GND. DIR1 pin is connected to RPi's pin 13(GPIO 27), PWM1 pin is connected to RPi's pin 15 (GPIO 22), DIR2 is connected to RPi's pin 16 (GPIO 23), PWM2 is connected to RPi's pin 18 (GPIO 24), and GND is connected to GND of Gyro.

Pins on the left side of the motor driver communicate with the battery source and motors. M1A is connected to the negative terminal and M1B is connected to the positive terminal of the Motor A (Left), M2A is connected to the negative terminal and M2B is connected to the positive terminal of the Motor B(Right). B+ and B- mentioned within the motor driver are connected to the positive and negative terminal of the Li-Ion Battery pack.

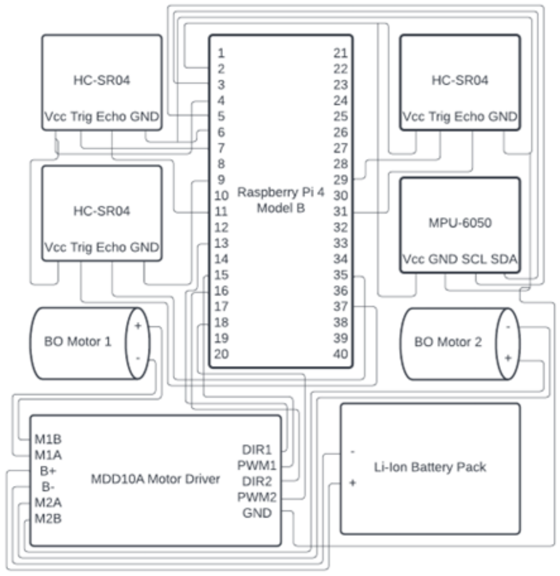

Fig. 3. Circuit diagram inside the Terrestrial Robot

C. Sensing

The equipped ultrasonic sensors (HC-SR04) [13] have two transducers each. One of which converts electronic pulses into ultrasonic waves, while the other one is a receiver that receives those ultrasonic waves and converts them back to electronic pulses. The time taken for each pulse is recorded to estimate the distance between the sensor and the corresponding object in front of it. Fig. 4. Shows the placement of the three different ultrasonic sensors placed at three different locations within the terrestrial robot.

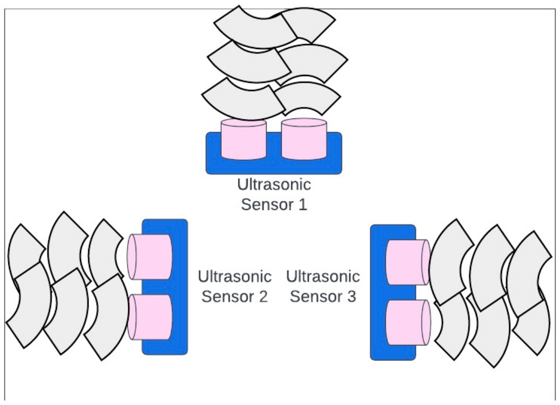

Fig. 4. Three different ultrasonic sensors placed at three different side of Terrestrial Robot

D. Equations

$$\frac{Resolution\ of\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage} \qquad (1)$$

Equation 1: Calculation of ADC Value

The resolution of ADC is considered as 1023 in the given equation, while the system voltage is considered 5 Volts, because the MPU6050 is connected to the 5V pin of the Raspberry Pi. The Analog Voltage will be measured by the sensor and will assist in calculating the ADC reading.

$$yaw = \frac{atan2(x,y)\times 180}{3.14} + 180 \qquad (2)$$

Equation 2: Calculation of yaw

The value of yaw is calculated using (2) and if the robot deviates more than 5 degrees, the speed of the opposite motor is adjusted to correct its value and orientation.

E. Working

Everything was built over a Terrestrial Robot chassis made up of PLA filament. This experiment was initiated by giving a 5 V of power to the Raspberry Pi 4 [14] with an external voltage of 11.1V given to the motor driver. We had initially set up the counter to 0 and the Terrestrial Robot keeps moving in a loop until the counter gets set to 4. The Ultrasonic Sensor 1 which is placed at the front face of the Terrestrial Robot detects the obstacle within its path and stores its distance from it in a CSV file. While the robot moves in a straight line, i.e. the paths detected by Ultrasonic Sensor 1, the Ultrasonic Sensor 2 and Ultrasonic Sensor 3 are placed on the left and right faces respectively to detect the obstacles within a range of their ultrasonic sensor and make those as their path for Terrestrial Robot movement. Assuming the obstacles are fixed so different sensors collect this information and stores it in a CSV file with three different set of columns namely, Ultrasonic Sensor 1, Ultrasonic Sensor 2, and Ultrasonic Sensor 3. On detecting the path by Ultrasonic Sensor 1, the robot moves the distance calculated for the obstacle, i.e., Total Distance until 40cm is left from the obstacle.

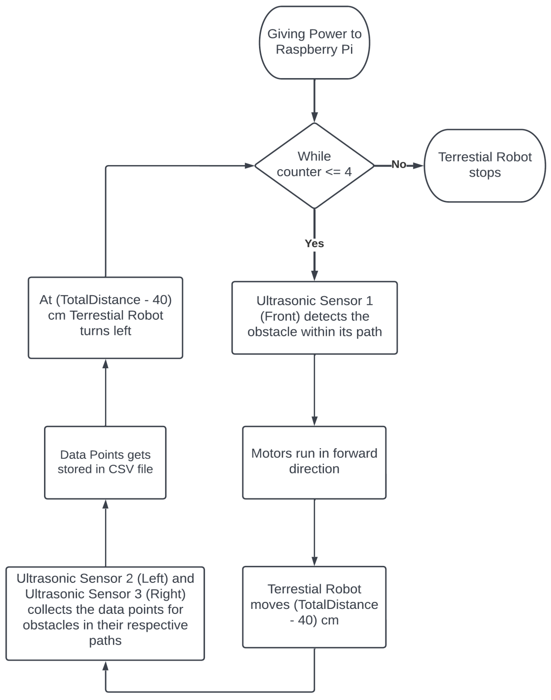
Giving Power to Raspberry Pi
While counter <= 4
No
Terrestial Robot stops
Yes
Ultrasonic Sensor 1 (Front) detects the obstacle within its path
Motors run in forward direction
Terrestial Robot moves (TotalDistance - 40) cm
Ultrasonic Sensor 2 (Left) and Ultrasonic Sensor 3 (Right) collects the data points for obstacles in their respective paths
Data Points gets stored in CSV file
At (TotalDistance - 40) cm Terrestial Robot turns left

Fig. 5. Flowchart for Terrestrial Robot Movement

The moment the robot reaches the specified distance, it makes a turn of 90 in the left direction, following the same looping mechanism until it reaches the unified obstacle distance. To make the turn, the left motor moves in the backward, and the right motors move in the forward direction respectively with a frequency of 75 Hz.

Note that, throughout this process, data information is collected by a terrestrial wheel robot which is then used to canvas the map for humanoid visuals.
Fig. 5. Described in the form of flowchart illustrates about the movement of Terrestrial Robot.

F. Algorithm

Algorithm 1: Mapping and Data Collection

```
1.  Trigger1 ← 4
2.  Echo1 ← 17
3.  Trigger2 ← 26
4.  Echo2 ← 19
5.  Echo3 ← 6
6.  Motor1Dir ← 27
7.  Motor2Dir ← 23
8.  Motor2Dir ← 23
9.  Motor1PWM ← 22
10. Motor2PWM ← 24
11. Trigger1 ← GPIO.OUT
12. Echo1 ← GPIO.IN
13. Trigger2 ← GPIO.OUT
14. Echo2 ← GPIO.IN
15. Trigger3 ← GPIO.OUT
16. Echo3 ← GPIO.IN
17. Motor1Dir ← GPIO.OUT
18. Motor2Dir ← GPIO.OUT
19. Motor1PWM ← GPIO.OUT
20. Motor2PWM ← GPIO.OUT
21. AzCal ← 0
22. GzCal ← 0
23. def distanceBySensor1
24.     GPIO.output(gpioTrigger1, True)
25.     sleep(0.00001)
26.     startTime ← time()
27.     stopTime ← time()
28.     while Echo1.IN == 0 do
29.         startTime ← time()
30.         end while
31.     while Echo1.IN == 1 do
32.         stopTime ← time()
33.         end while
34.     timeElapsed ← stopTime − startTime
35.     distanceBySensor1 ← (timeElapsed ∗34300)/2
36.     return distanceBySensor1
37. def distanceBySensor2
38.     GPIO.output(gpioTrigger2, True)
39.     sleep(0.00001)
40.     startTime ← time()
41.     stopTime ← time()
42.     while Echo2.IN == 0 do
43.         startTime ← time()
44.         end while
45.     while Echo2.IN == 1 do
46.         stopTime ← time()
47.         end while
48.     timeElapsed ← stopTime − startTime
49.     distanceBySensor2 ← (timeElapsed ∗ 34300)/2
50.     return distanceBySensor2
51. def distanceBySensor3
52.     GPIO.output(gpioTrigger1, True)
53.     sleep(0.00001)
54.     startTime ← time()
55.     stopTime ← time()
56.     while Echo3.IN == 0 do
57.         startTime ← time()
58.         end while
59.     while Echo3.IN == 1 do
60.         stopTime ← time()
61.         end while
62.     timeElapsed ← stopTime − startTime
63.     distanceBySensor3 ← (timeElapsed ∗ 34300)/2
64.     return distanceBySensor3
65. dataFile ← open('MapData.csv',' w+')
66. with dataFile:
67. while True do
68. distanceCalculatedBySensor1=distanceBySensor1()
69. distanceCalculatedBySensor2=distanceBySensor2()
70. distanceCalculatedBySensor3=distanceBySensor3()
71. dataFile.writeRow
    (distanceCalculatedBySensor1,
    distanceCalculatedBySensor2,
    distanceCalculatedBySensor3)
72.     z ← readMPU(ACCELZ)
73.     AccelerationZ ← z/16384 − AzCal
74.     gZ ← readMPU(GY ROZ)
75.     GyroZ ← gZ/131 − GzCal
76.     if distanceCalculatedBySensor1 ≤ 40 then
77.         if abs(GyroZ) ≤ 90 then
78.             Motor1Dir ← LOW
79.             Motor2Dir ← HIGH
80.             dataFile.writeRow('left', 'left', 'left')
81.         end if
82.             Motor1Dir ← HIGH
83.             Motor2Dir ← HIGH
84.         end if
85. end while
```

The Mapping and Data Collection algorithm consists of various sets of explicit and unambiguous finite steps which, were used during the experimental test of the terrestrial robot. As explained in section 2B various components are used to achieve the utilization of the mapping and data collection algorithm that is embedded as a part of running Terrestrial Robot. The algorithm defines the trig and echo pins through Trigger1, Trigger2, Trigger3, Echo1, Echo2, and Echo3 variables where Trigger1, Trigger 2, and Trigger3 variables are set to GPIO.OUTPUT and Echo1, Echo2, and Echo3 are set to GPIO.INPUT. Similarly, the pins of the motor driver, i.e., Motor1Dir, Motor1PWM, Motor2Dir, Motor2PWM. The distance between the robot and obstacles is calculated using ultrasonic sensors. The ultrasonic wave is sent out from trig for a fraction of a second, and it is received at echo. The timestamps for sending and retrieval are recorded and subtracted to calculate the time elapsed. The speed of sound is taken as 343m/s, so the distance is calculated by multiplying the speed by the time elapsed and dividing the result by 2. The same algorithm is repeated for ultrasonic sensor 2 and ultrasonic sensor 3, respectively. A CSV (Comma Separated Values) file is created for storing the data points. The distances are stored under the columns "Ultrasonic Sensor 1", "Ultrasonic Sensor 2", and "Ultrasonic Sensor 3". MPU6050's [15] readMPU function is used to read the accelerometer and gyro values for Z-axis. Once the robot reaches a distance of 40cm from the obstacle, it will check if the absolute value of the gyro on the Z-axis is less than or equal to 90. Until the value of the gyro turns out to be 90 degrees, it will keep turning left. Additionally, the CSV file will be appended with rows stating the change of direction. If the distance between the robot and the obstacle is greater than 40cm, it will keep moving in the forward direction.

G. *Map Generation*

The data points collected by the Terrestrial Robot which gets stored in a CSV file as explained in section 2E are used to fulfill the process of *mapping*. The Ultrasonic Sensor 1 (Front), Ultrasonic Sensor 2 (Left), and Ultrasonic Sensor 3 (Right) plots the *line* in front, left, and right position. Hence with this, humans find the *graphical map plotting*. The arena was initially plotted using Tkinter library [20] Canvas, and then the same was plotted in Figma for clarity.

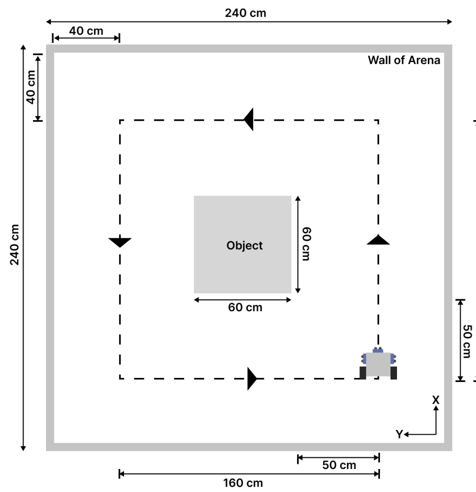

Fig. 6. A schematic of the experimental setup and the path charted by the robot with dimensions in cm

A square arena of dimension 240 cm x 240 cm was developed, and a square obstacle of dimension 60 cm x 60 cm was placed at the center. Initially, the distance recorded by sensors 1, 2, and 3 was 200cm, 200cm, and 40cm from the robot to the front, left and sidewall respectively. The arena is shown in fig 6.

III. Experimental Results

A. *Data Collection*

The data information used for the autonomy of a terrestrial robot is prepared under the process of *data collection*. The data points are collected using three different ultrasonic sensors placed on three sides of the chassis. Data points are stored in the CSV file with three different column sections namely *Ultrasonic Sensor 1*, *Ultrasonic Sensor 2*, and *Ultrasonic Sensor 3*. The data points get stored in the cartesian plane [21] within the *X-axis*, *-X-axis, Y-axis*, and *-Y-axis* with 4 degrees of freedom.

Note that, the data collected in the CSV file is a mathematical map for the Terrestrial Robot which it uses to self-learn the autonomous path.

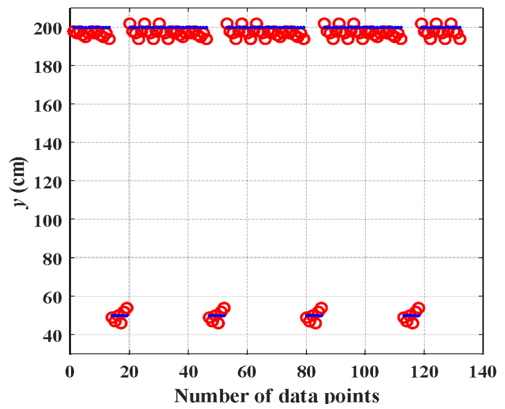

Fig. 7. Representation of collective data points collected from four sides.

Fig 7. explains about the data collected from four sides of the arena is represented collectively in the cartesian plane in the form of a constant line passing through y=200 and y=50. The y=200 represents the distance of the side arena wall from the robot taken by the side sensor and the y=50 represents the distance of an object from the robot taken by the same side sensor. These data points are represented through the side sensors which get overlapped by the other sensor's readings.

Here, fig 7. represents the reading taken by the side sensor of the wall placed at 200cm from the robot while the robot keeps moving. On encountering the object placed 50cm from the robot is represented at y=50cm.
These data points are also used to make the humanoid graphical visual map to understand the path in a better way.

B. *Simulation*

The map environment was manually generated by taking theoretical data points (Hypothetical) to see the effects of the multi-ultrasonic sensors as shown in fig 8. Problems like phantom points could be seen due to specular reflections of obstacles that get rectified by running the robot on the same path. Different colors on the map indicate that the robot ran at different paths indicated by the different symbolic line. Limitations due to sensors disabling at corners, it does not capture the data points at the corners.

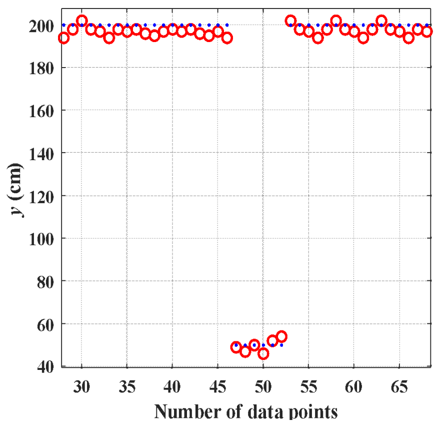

Fig. 8. Zoomed view of data points collected from one side.

C. *Real Environment*

The arena was made with the help of cardboard, representing the walls for the experimentation run of a Terrestrial Robot. The Terrestrial Robot was moved manually parallel to the walls, as it favored the sensor readings. Data points were collected and they were passed through the filtration process manually to get the best results for mapping which could be seen in fig 9.

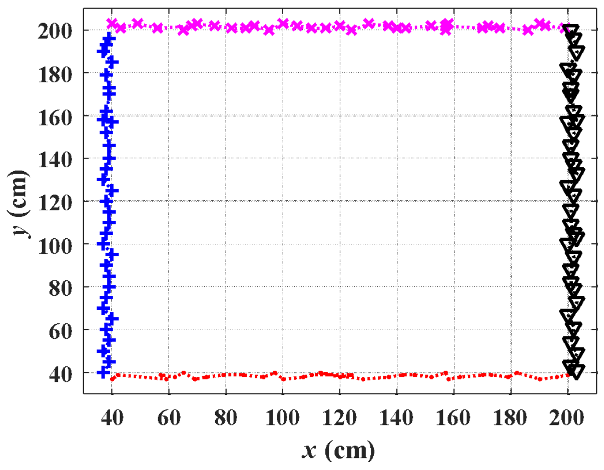

Fig. 9. Actual path traced by the Terrestrial Robot

The error recorded throughout this experimentation process in consideration of simulation and the real environment was 2.016*%,* hence the error at a particular point was calculated in the following way: 100 x (Ideal data - Collected Data) / Ideal data. Later, a mean percentage error was calculated for the data collected by each sensor.

The percentage error turned out to be 1.5167%, 2.1667%, and 2.3636% for sensors 1, 2, and 3 respectively, which has been represented in fig 10. The X-axis denotes the used ultrasonic sensor, while the Y-axis denotes the percentage error.

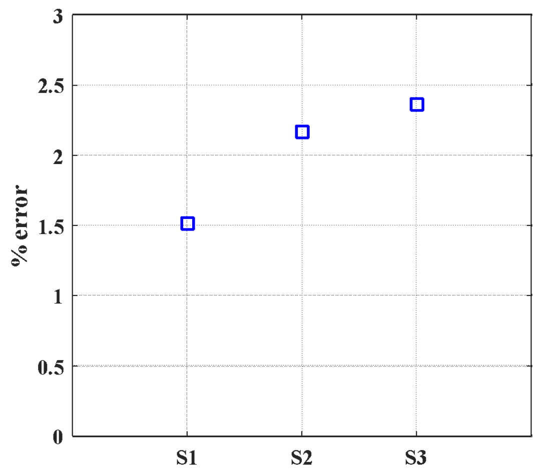

Fig. 10. Mean error percentages for each of the three sensors.

TABLE 1: Cost comparisons between different sensors available for mapping and localization

Sensors	**Cost (in Rupees)**
LIDAR	9000
Depth Camera	40000
Sweep	27427
HC SR-04 Ultrasonic	150

Table 1, denotes the price comparisons between various sensors that are the best fit for the mapping and localization. The ultrasonic sensor used in terrestrial robot is least expensive which makes this experiment and setup cost effective with the optimized algorithm.

IV. Conclusion

In this paper, we represented the algorithmic, and cost-effective solution by developing the optimized algorithm, by ultrasonic sensors within the terrestrial robot. The proposed approach is based on four fundamental processes: sensing, localization, map generation, and humanoid visualization. The Terrestrial Robot runs alongside the arena walls, collecting the data points, and generating the map with the gathered information.

References

[1] Jung, S., Kim, J. & Kim, S. Simultaneous localization and mapping of a wheel-based autonomous vehicle with ultrasonic sensors. Artif Life Robotics 14, 186 (2009). https://doi.org/10.1007/s10015-009-0650-9.

[2] S. K. A. Nair, S. Joladarashi and N. Ganesh, "Evaluation of ultrasonic sensor in robot mapping," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 638-641, DOI: 10.1109/ICOEI.2019.8862659.

[3] A. Kurz, "Constructing maps for mobile robot navigation based on ultrasonic range data," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26, no. 2, pp. 233-242, April 1996, DOI: 10.1109/3477.485835.

[4] G. C. Anousaki and K. J. Kyriakopoulos, "Simultaneous localization and map building for mobile robot navigation," in IEEE Robotics &

Automation Magazine, vol. 6, no. 3, pp. 42-53, Sept. 1999, DOI: 10.1109/100.793699.

[5] Zou Yi, Ho Yeong Khing, Chua Chin Seng, and Zhou Xiao Wei, "Multi-ultrasonic sensor fusion for mobile robots," Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511), 2000, pp. 387-391, DOI: 10.1109/IVS.2000.898374.

[6] Y. Liu et al., "Mobile Robot Localisation and Navigation Using LEGO NXT and Ultrasonic Sensor," 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2018, pp. 1088-1093, DOI: 10.1109/ROBIO.2018.8665350.

[7] G. Oriolo, G. Ulivi and M. Vendittelli, "Real-time map building and navigation for autonomous robots in unknown environments," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 28, no. 3, pp. 316-333, June 1998, DOI: 10.1109/3477.678626.

[8] M. F. Norazman and N. M. Thamrin, "Landmark scanning by using infrared sensor for simultaneous localization and mapping application," 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA), 2018, pp. 145-149, doi: 10.1109/CSPA.2018.8368702.

[9] R. W. Wall, J. Bennett, and G. Eis, "Creating a low-cost autonomous vehicle," IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02, 2002, pp. 3112-3116 vol.4, doi: 10.1109/IECON.2002.1182894.

[10] J. Lim, S. Lee, G. Tewolde, and J. Kwon, "Indoor localization and navigation for a mobile robot equipped with rotating ultrasonic sensors using a smartphone as the robot's brain," 2015 IEEE International Conference on Electro/Information Technology (EIT), 2015, pp. 621-625, doi: 10.1109/EIT.2015.7293407.

[11] I. A. R. Ashokaraj, P. M. G. Silson, A. Tsourdos, and B. A. White, "Robust Sensor-Based Navigation for Mobile Robots," in IEEE Transactions on Instrumentation and Measurement, vol. 58, no. 3, pp. 551-556, March 2009, doi: 10.1109/TIM.2008.2005266.

[12] S. Kodagoda, E. A. S. M. Hemachandra, P. G. Jayasekara, R. L. Peiris, A. C. De Silva, and R. Munasinghe, "Obstacle Detection and Map Building with a Rotating Ultrasonic Range Sensor using Bayesian Combination," 2006 International Conference on Information and Automation, 2006, pp. 98-103, doi: 10.1109/ICINFA.2006.374159.

[13] Morgan, Elijah J. "HC-SR04 ultrasonic sensor." (2014): 1-5.

[14] Upton, Eben, and Gareth Halfacree. Raspberry Pi user guide. John Wiley & Sons, 2014.

[15] Fedorov, D. S., et al. "Using of measuring system MPU6050 for the determination of the angular velocities and linear accelerations." Automatics & Software Enginery 11.1 (2015): 75-80.

[16] Collis, R. T. H. "Lidar." Applied optics 9.8 (1970): 1782-1788.

[17] Hawkins, Douglas M. Identification of outliers. Vol. 11. London: Chapman and Hall, 1980.

[18] Durrant-Whyte, Hugh, and Tim Bailey. "Simultaneous localization and mapping: part I." IEEE robotics & automation magazine 13.2 (2006): 99-110.

[19] Elfes, Alberto. "Sonar-based real-world mapping and navigation." IEEE Journal on Robotics and Automation 3.3 (1987): 249-265.

[20] Lundh, Fredrik. "An introduction to tkinter." URL: www.pythonware.com/library/tkinter/introduction/index. htm (1999)

[21] Bill, E. Gordon. "CA Scott, Cartesian Plane Geometry. Part I.: Analytical Conics." Bulletin of the American Mathematical Society 15.10 (1909): 507-510.

[22] J. Salmerón-Garcı´a, P. Íñigo-Blasco, F. Dı´az-del-Rı´o and D. Cagigas-Muñiz, "A Tradeoff Analysis of a Cloud-Based Robot Navigation Assistant Using Stereo Image Processing," in *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 444-454, April 2015, doi: 10.1109/TASE.2015.2403593.

[23] W. Ju, ‘Application of autonomous navigation in robotics’, *Journal of Physics: Conference Series*, τ. 1906, τχ. 1, σ. 012018, Μαΐου 2021.