

RESEARCH

Open Access



A low-cost robotic system for simultaneous localization and mapping

Ayman Hamdy Kassem^{1*}  and Muhammad Asem¹

*Correspondence:
akassem@cu.edu.eg

¹ Aerospace Department, Faculty
of Engineering, Cairo University,
Cairo, Egypt

Abstract

This paper presents a low-cost system for simultaneous localization and mapping (SLAM) for unknown indoor environments. The system is based on a low-cost mobile-robot platform. The low-cost mobile robot is designed and fabricated in our control laboratory. The Rao-Blackwellized particle filter algorithm is used for SLAM computations, Xbox 360 Kinect module is utilized for stereo-camera imaging, and a Linux-based microcomputer (Raspberry Pi3) was used as the main onboard processing unit. An Arduino board is used to control the DC motors for mobile robot wheels. Raspberry Pi unit was wirelessly connected to a ground station machine that processes the information sent by the robot to build the environment map and estimate its pose. ROS (Robot Operating System) is used for map visualization, data-handling, and communication between different software nodes. The system has been tested virtually on a simulator and in real indoor environments and has successfully identified objects greater than 30 cm × 30 cm × 30 cm and added it to the map. It also shows promising capability to work autonomous missions independently without aid from any external sensors and with a fraction of the cost of similar systems based on Lidars.

Keywords: SLAM, ROS, Raspberry Pi, Probabilistic robotics, Particle filters, Mobile robot, Arduino, Self-driving cars

Introduction

Simultaneous localization and mapping (SLAM) problem is the computational process of creating a map of an unknown environment using mobile robot(s). The robot should also figure out its location in the environment or the created map. SLAM problem had attracted researchers' attention in the recent years with the rise of self-driving cars, Mars exploration, deep sea Exploration, etc. SLAM enables the robot to perform its autonomous mission independently without aid from any external sensors or systems like GPS (Global Positioning System) or IPS (Indoor Positioning System). SLAM utilizes its own onboard sensors to have a good environment perception to navigate easily.

The mapping task in SLAM is a crucial problem in autonomous navigation which depends on maps for goal setting and path execution. Maps are built by combining robot poses (positions and orientations) with relative scans of the seen landmark and adding it all to the global map frame of reference. The localization task is a challenge that arises when there is no localization source inside the environment like a GPS or IPS. In that

case, you will need a map with landmarks to determine your location. It is more like an egg-chicken problem where mapping needs localization and vice versa. Repeated passes in the target scene can improve both mapping and localization of the SLAM system. Different approaches and algorithms are used to solve the SLAM problem [1–3] and we have selected to use a Rao-Blackwellized particle filter [4] in this work. Solving the SLAM problem depends on distance measurements. This measurement task can be accomplished by utilizing either high-cost, precise laser scanners (Lidars) or less expensive but less precise stereo-cameras. However, for commercial applications in shopping malls and similar facilities, this work will focus on using Kinect stereo cameras which offer a satisfactory balance between cost and accuracy. Our objective is to construct an affordable robotic system that utilizes off-the-shelf components and employs the most stable SLAM algorithms available.

Literature review of the SLAM problem

Chatila and Laumond [5] and Smith et al. [6], in the eighties of the last century, were the first to introduce the concept of concurrently estimating the pose (position + orientation) while building the map which is known later as simultaneous localization and mapping (SLAM) and is referred to as concurrent localization and mapping (CML) in some publications. During this period the probabilistic approaches were at their beginning to be introduced to Artificial Intelligence (AI) and Robotics. Several key papers were published that establish a statistical basis: for describing the relationship between landmarks and manipulating geometric uncertainty like the work done by Durrant-Whyte [7, 8] and for utilizing the ultrasonic sensors and Kalman filter in navigation as done by Crowley [9].

Literature has revealed a variety of solutions to this problem which can be divided into filtering and smoothing approaches. Filtering is intended to solve the online SLAM problem as it estimates the current robot pose and the environment map, and then it updates and refines the state belief (estimate) by fusing the new available state measurements. Kalman [10] and particle filters [11, 12] form two of the most popular techniques used to solve the online SLAM problem.

In contrast, smoothing techniques which rely on least square error minimization, estimate the full trajectory of the robot from the full set of measurements [13–15] to solve the full SLAM problem.

Basically, the localization and mapping problems were addressed and studied independently assuming that the map is given in the case of the localization problem [16, 17] and the pose is given in the case of the mapping problem which is known as mapping with known poses problem.

Realizing simultaneous localization and mapping (SLAM) problem requires dealing with practical considerations like sensors and actuator noise in addition to the uncertainties associated with the mathematical models due to assumptions made in derivations. These noises and uncertainties are accumulated with time leading to a non-reliable solution for the problem.

Filters are proposed as an efficient and online solution to overcome the error accumulation process. Different filters are used based on the Bayes filter and Bayesian estimation [18] by estimating the current state depending on the last state and observations.

Extended Kalman filter (EKF) was used as an early solution to solve the SLAM problem by formulating the problem under the state space model using a state vector to represent the state of the robot and the environment, and a covariance matrix to express the uncertainty associated with the state vector.

The work presented by Leonard showed the application of extended Kalman filter EKF in the problem of navigation in a known environment by tracking known geometric beacons which formed a good motivation for the later work in estimating both the pose and the map concurrently.

Randall et al. [19] introduced the concept of the stochastic map, i.e., a map that shows the uncertainties and the relationship between its entities (landmarks). It used probabilistic approaches for the estimation of this stochastic map; it used the Kalman filter as an initial solution and then introduced the extended Kalman filter to deal with the nonlinearities associated with the observation and the motion models.

The work done by Castellanos et al. [3] evaluates the influence of correlation between map entities on the process of robot relocation and global map building of the environment of a mobile robot navigating in an indoor environment. It uses the EKF filter approach besides a probabilistic model to represent the uncertainties in the state. The filter was used to incorporate the onboard sensor measurements (like laser range radar) with the predictions from the dynamical model to have a good belief about the current state.

Particle filters Fox et al. [1, 16] were the first to use particle filter for localization then it was used later after enhancement (Rao-Blackwellized particle filter) for estimating the pose and the map concurrently. Monte Carlo localization is named after particle filter implementation in localization problems. Their work presented the solution of the localization problem based on probabilistic methods; it aims to get the probability distribution over the robot position. It represents the probability density by maintaining a set of samples that are randomly drawn from it, contrary to the Kalman filter which is based on parametric Gaussian distributions. Monte Carlo localization (MCL) can represent arbitrary non-parametric distributions (i.e., do not have mean and variance) for localization across a map without knowledge of the robot's starting location.

Rao-Blackwellized particle filter (FastSLAM) [4, 12, 20–22] is a variant of the particle filter algorithm used in state estimation problems, such as simultaneous localization and mapping (SLAM). It is a probabilistic algorithm that combines particle filtering with the Rao-Blackwellized technique, which is a method for factoring a joint probability distribution into a product of conditional probability distributions. It is used to solve the SLAM problem by dividing the problem into a robot localization problem and landmark estimation problems (i.e., estimation of each landmark location alone) that are conditioned on the robot pose estimate.

FastSLAM uses a modified particle filter (Rao-Blackwellized particle filter) [23, 24] for estimating the posterior over the robot path, each particle possesses a number of K Kalman filters that estimate K landmark locations. The advantage here is that a 2×2 Kalman filter (as it estimates the x and y locations of the landmark) is used per landmark instead of using a huge Kalman filter matrix as in EKF SLAM that contains the robot state and all landmarks' states in a single covariance matrix.

Recent research as the work of Xiaobin Xu et al. [25] concentrates on Lidar sensors for improving the accuracy of positioning of the robot using an Adaptive Federated Kalman Filter. The cost of Lidars is approximately 15 times higher than that of the Kinect camera. While Lidar offers greater distance measurement accuracy, the work done by Ahmad el. Al. [26] proves that we can get acceptable positioning accuracy with Kinect Camera for commercial applications, such as cleaning and garbage collection, in shopping malls and similar facilities, where higher levels of positioning accuracy are not essential.

The mobile base used in this work is a differential drive robot which is the simplest form of mobile robots which uses two wheels. Differential drive robots are commonly used in robotics research [27, 28], and education, as well as in industrial applications such as material handling and warehouse automation. They are relatively simple and inexpensive to build and can be highly maneuverable in tight spaces.

Scientific background

Problem definition

Simultaneous localization and mapping (SLAM) refers to the problem of estimating the pose of a robot, and the map of an unknown environment surrounding it concurrently using onboard sensors only [2].

It is considered a chicken-and-egg problem, as a map is needed for localization and localization is needed for mapping. This is a key problem in probabilistic robotics and autonomous navigation as it estimates the robot's state and the environment map.

SLAM problem can be formulated as:

Given:

1. The robot's controls: which command its motion, $u_{1:T} = \{u_1; u_2, \dots, u_T\}$ with uncertainty to a limit in their execution as they are transmitted via noisy hardware actuators.
2. The observations: are the information we get about the environment and robot state $z_{1:T} = \{z_1; z_2, \dots, z_T\}$ through the robot sensors which are inherently noisy.

Required:

1. A map of the environment m that describes the locations of the needed
2. The path of the robot $x_{0:T} = \{x_0, x_1, x_2, \dots, x_T\}$

Which can be summarized in the following term of probability distribution:

$$p(x_{0:T}, m \mid z_{1:T}, u_{1:T}) \quad (1)$$

The diagram shows the probability distribution $p(x_{0:T}, m \mid z_{1:T}, u_{1:T})$ with arrows pointing to its components: $x_{0:T}$ is labeled 'distribution', m is labeled 'path', $z_{1:T}$ is labeled 'map', \mid is labeled 'given', $z_{1:T}$ is labeled 'observations', and $u_{1:T}$ is labeled 'controls'.

This probability distribution is the SLAM problem target which seeks to find the path and the map of the robot environment given the observation and the controls.

Figure 1 represents the problem in a graph model that shows the connection between the unknown and the observed nodes. x_t is the robot current state, u_t is the current control input, z_t current observation from the robot's point of view and m is the map or landmark's locations.

Control inputs (known) affect the current state (unknown) that is affected by the previous state. The state affects the observation model as the robot's observation for the landmark depends on its location and affects its observation.

Mobile robot kinematics

Mobile robot introduced in this research is a differential drive robot that uses two wheels to move and turn. The robot's motion is controlled by varying the relative speeds of the two wheels. By driving the wheels in opposite directions at the same speed, the robot will move forward or backward. By driving the wheels at different speeds in opposite directions, the robot will turn in place. The robot can be modeled mathematically by two systems of equations (Eqs. 2 and 3) [27]. System of Eq. 1 relates the angular velocity W of the robot, and the linear velocity command v to the robot states x, y, q (as shown in Fig. 2).

$$\dot{x} = v \cdot \cos(\theta), \dot{y} = v \cdot \sin(\theta), \dot{\theta} = \Omega \quad (2)$$

System of Eq. 2 relates the control inputs of the robot v and q to each robot DC motor angular velocity Q_r for the right motor and Q_l for the left one, given the wheel diameter of the robot R and the width of the robot L as shown in Fig. 3.

$$\Omega_r = \frac{2v + \Omega L}{2R}, \Omega_l = \frac{2v - \Omega L}{2R} \quad (3)$$

Odometry calculation

Wheel encoders are used to estimate the position and the heading of the robot incrementally overtime from a starting location. Optical wheel encoders were used. It divides each revolution of the N divisions and uses a light source and a light sensor to detect how many pulses are generated (ticks) by cutting the light source. Cutting or ticking N

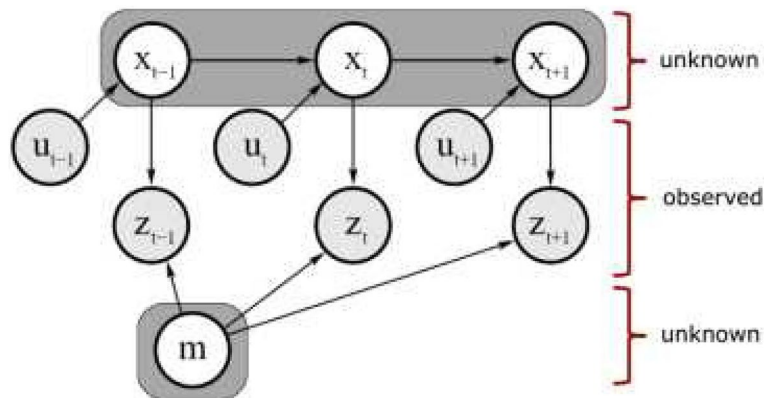


Fig. 1 Graph model of SLAM problem $p(x_{0:T}; I_{1:M} | z_{1:T}; u_{1:T})$

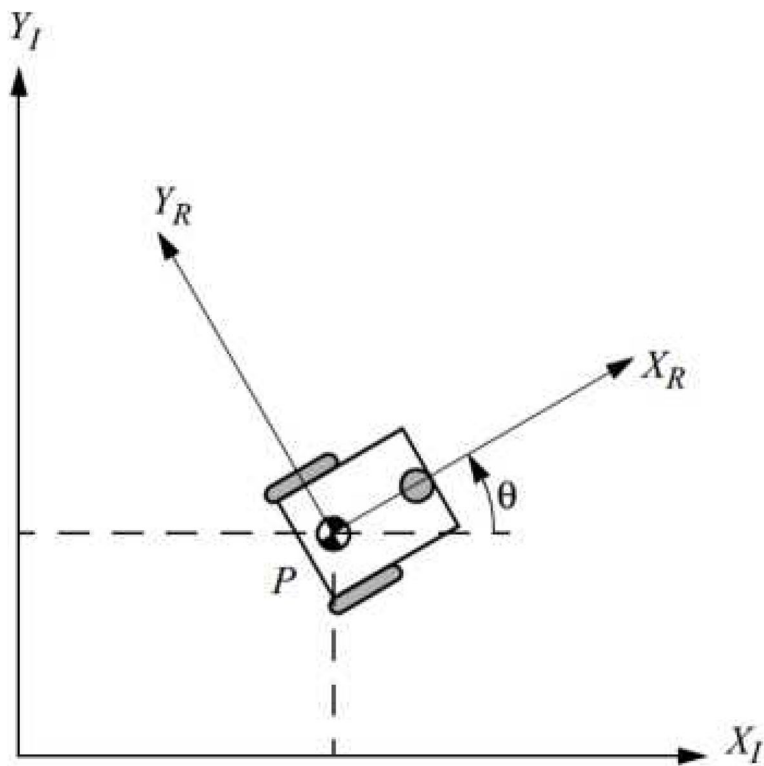


Fig. 2 Body and global frames of reference of the robot

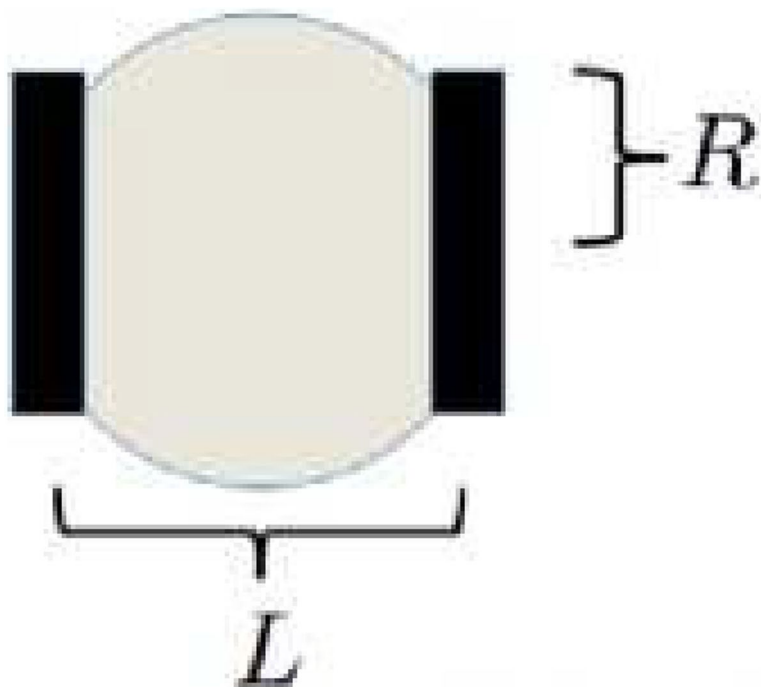


Fig. 3 Mobile robot wheel and width dimensions

times indicates a full revolution. Equation 4 calculates the distance traveled D during time Dt .

$$\Delta \text{tick} = \text{tick}^{t+1} - \text{tick}^t, D = 2\pi R \frac{\Delta \text{tick}}{N} \quad (4)$$

Figure 4 shows the distance traveled by each wheel, right D_r and left D_l , and the center of mass D_c .

The distance traveled by the center of mass is considered as the distance traveled by the robot. Equation 5 calculates the distance traveled by the center of mass given the distance covered by each wheel after the encoder readings.

$$D_c = \frac{D_l + D_r}{2} \quad (5)$$

The absolute position from the starting point and the heading can be calculated incrementally using Eq. 6. So, the feedback of the states (x, y, q) can be measured using the wheel encoders.

$$x^{t+1} = x^t + D_c \cdot \cos(\theta), y^{t+1} = y^t + D_c \cdot \sin(\theta), \theta^{t+1} = \theta^t + \frac{D_r - D_l}{L} \quad (6)$$

Xbox Kinect Stereo Camera

A single camera can't infer the depth directly, as all the points on the same line of sight with different depths are projected into a single point on the image plane. Using two cameras as in Xbox Kinect enables us to estimate the depth of the points. Calculating the depth of a point requires finding the same point in the two pictures, taken from the two stereo cameras, which is a computationally involved procedure with many image processing sub-functions such as rotation, translation, and rectification [26, 29, 30].

Rao-Blackwellized particle filter (FastSLAM)

Particle filter [16] can be used for solving the SLAM problem given the observations and the control commands if the problem dimensions are relatively low. As the landmark number increases the problem dimensions increases which in turn affects the particle

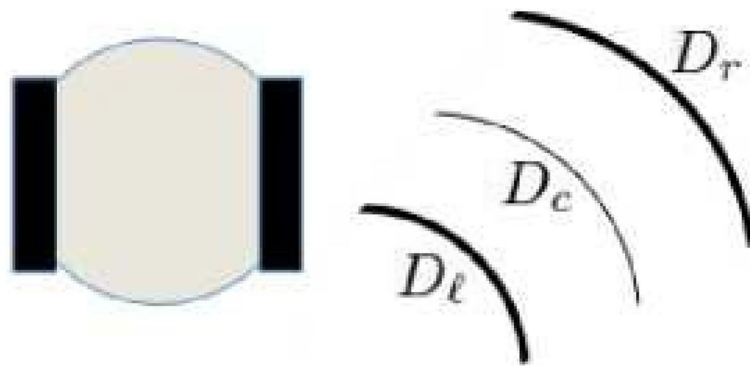


Fig. 4 Right D_r and left D_l distance traveled by each wheel of the differential drive robot, D_c is the distance traveled by the center of the robot

filter performance. A factorized solution was introduced [11] to avoid the dimensionality problem by solving the problem using two methods in conjunction, particle filter and EKF (Extended Kalman Filter).

The probability distribution in 1 can be factorized to

$$p(x_{0:T}, l_{1:M} | z_{1:T}, u_{1:t}) = p(x_{0:T} | z_{1:T}, u_{1:t}) p(l_{1:M} | x_{0:T}, z_{1:T}) \quad (7)$$

By referring to the graph model in Fig. 1 the conditional independence between landmarks' position could be deduced easily as the following:

$$p(l_{1:M} | x_{0:t}, z_{1:t}) = \prod_{i=1}^M p(l_i | x_{0:t}, z_{1:t}) \quad (8)$$

Equation 7 could be written as:

$$p(x_{0:T}, l_{1:M} | z_{1:T}, u_{1:t}) = p(x_{0:T} | z_{1:T}, u_{1:t}) \prod_{i=1}^M p(l_i | x_{0:t}, z_{1:t}) \quad (9)$$

The path posterior $p(x_{0:T} | z_{1:T}, u_{1:t})$ can be solved using the particle filter efficiently as it has low dimensions related to the robot pose, only $[x, y, q]$, while map posterior $\prod_{i=1}^M p(l_i | x_{0:t}, z_{1:t})$ could be solved by only 2×2 EKF for each landmark to estimate x and y positions of the landmark. So, the problem could be solved after factorization by exploiting the advantage of the particle filter and the EKF in conjunction.

Methods/experimental

The intended use of the proposed robotic system is to address the SLAM problem, which involves the robot navigating its environment while the stereo-camera captures images of the landmarks and calculates their distances. The fast-SLAM algorithm is employed to create an initial map and approximate the robot's location. As the robot continues to move and gather additional data, the map and location estimates are refined, leading to a final version after several iterations.

Figures 5 and 6 show the system hardware. The figures show Xbox Kinect, Raspberry Pi board, the motors micro-controller, the main LiPo battery on the middle layer of the robot as well as the DC motors with their encoders. They also show the two caster wheels (front and back) and the main robot wheels are on the bottom layer of the robot. The robot is made from low-cost plywood and assembled using bolts and nuts in addition to spacers. The main wheels are designed and made from acrylic. The structure is custom made in our lab and was designed on SolidWorks software and manufactured using laser cutting technique. Figure 7 shows the dimensions of the robot in millimeters (mm).

Figure 8 shows the connections schematic between different hardware components on the robot and the command station.

Software implementation

ROS (Robot Operating System) is a Linux-based meta-operating system initiated by Stanford [31]. ROS provides a collection of libraries and tools that simplify the development of complex robotic systems. It provides a distributed computing architecture that enables multiple software components to communicate and work together. ROS

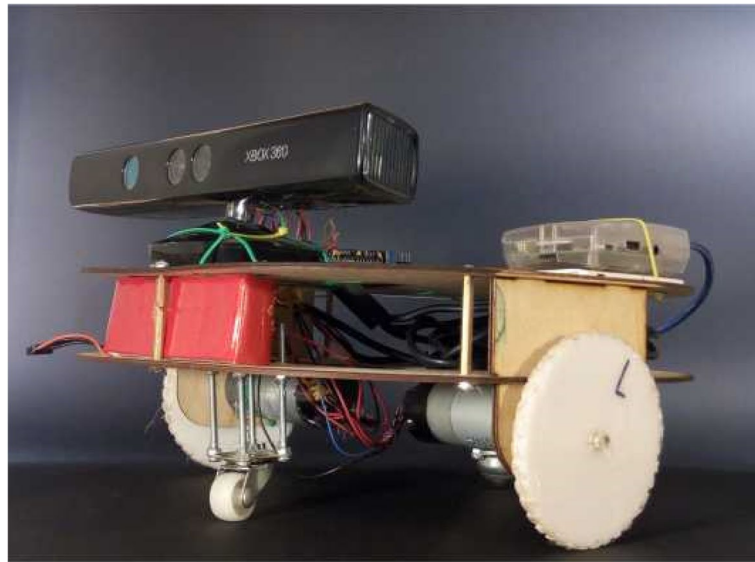


Fig. 5 SLAM system hardware (sideview)

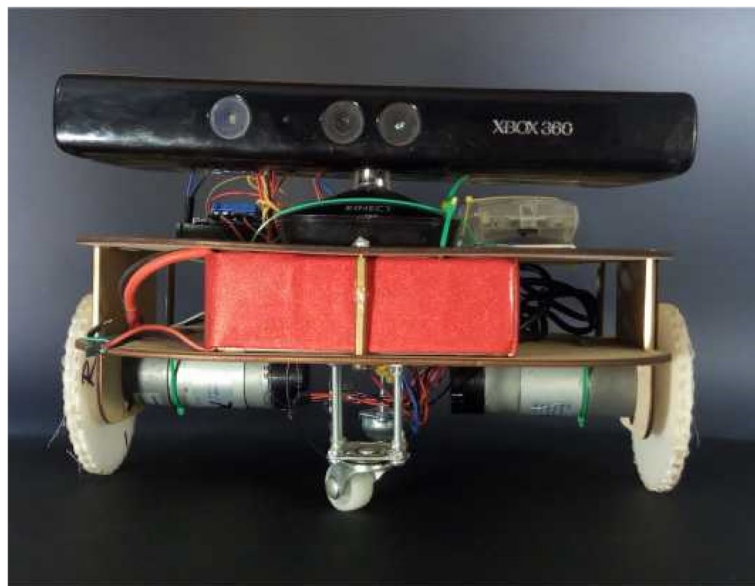


Fig. 6 SLAM system hardware (front view)

nodes are the basic building blocks of ROS applications, and they can communicate with each other using a publish/subscribe messaging system [32]. It was used as the main software framework to handle different software parts in the form of nodes. Nodes are programmed individually and connected under the ROS network of nodes to execute a certain task. Nodes can be written in different programming languages and on different hardware platforms. Communication between nodes over different hardware platforms can be done wirelessly over a WIFI router using HTTP protocols.

Nodes are mainly run on three hardware platforms as shown in Fig. 9.

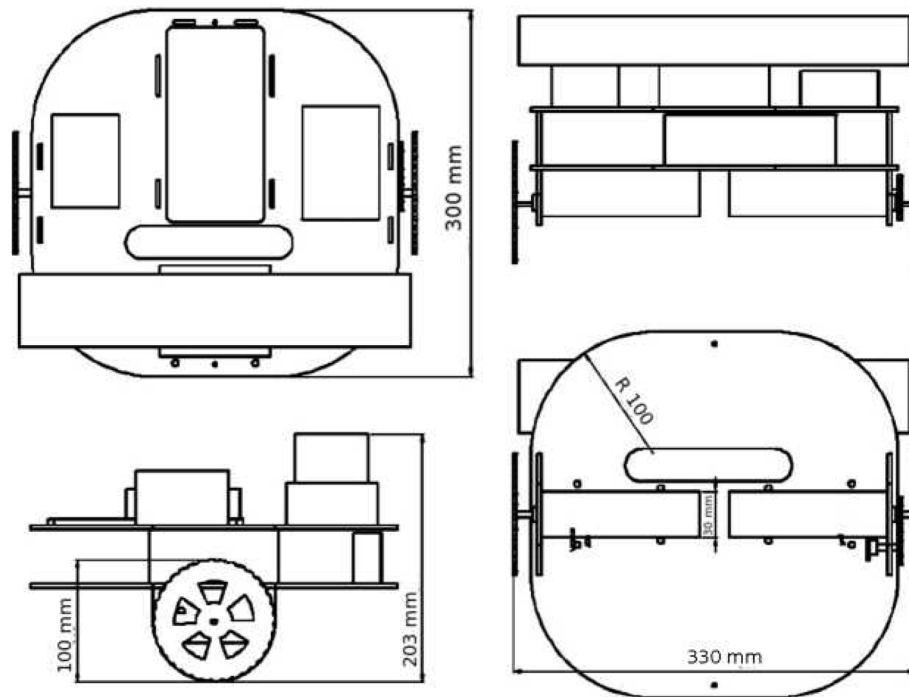


Fig. 7 Robot dimensions

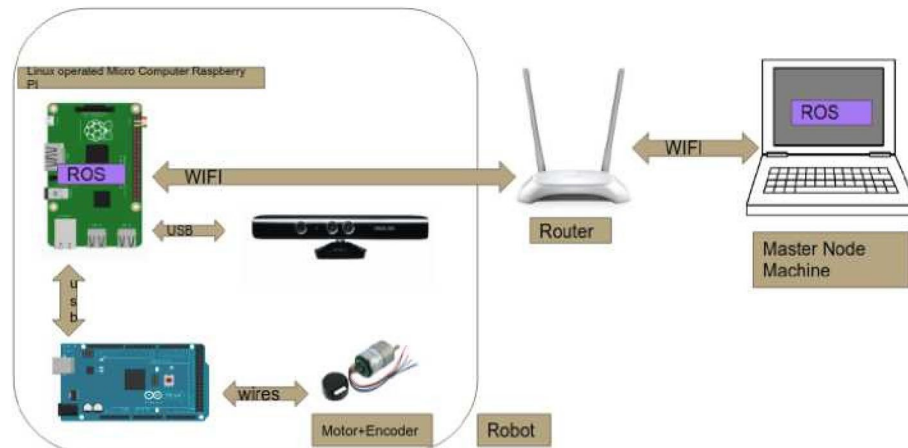


Fig. 8 Hardware connections schematic

1. Command station (laptop):

- (a) Command station is a laptop equipped with 4 cores Intel core i5 @2.5 GHz and a 4 GB RAM running Linux Ubuntu 16.04 LTS. It is responsible for:
- (b) Running the main ROS node (roscore) that handles and organizes the work between different software nodes.

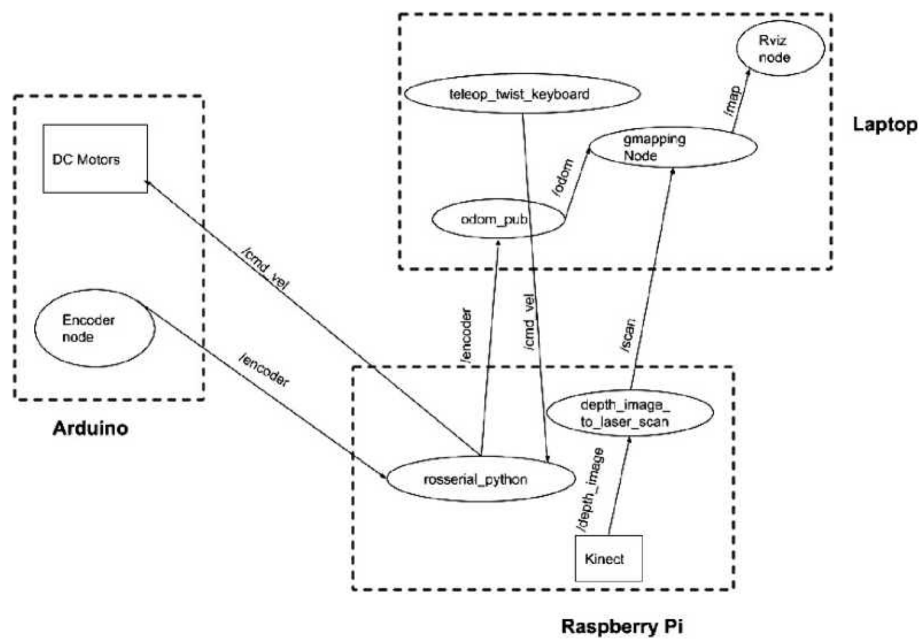


Fig. 9 ROS nodes

- (c) Running teleoperation node (teleop_twist_keyboard), which translates the key-board commands to a velocity and directions command over “/cmd.vel” topic.
- (d) Processing the encoder messages received from Arduino to calculate the robot odometry under “odom” topic using odom_pub node that will be used later by the mapping node.
- (e) Running the mapping node which generates the occupancy grid map, given the odometry readings and laser scans from the Xbox Kinect sensor.
- (f) Running visualization node (Rviz) that shows the map, the odometry, and all the processed information in a graphical form.

2. Onboard main processing computer:

Raspberry Pi 3 model B is the main onboard computer. It is equipped with Cortex-A53 ARM processor with 4 cores @1.2GHz and 1GB RAM in addition to supporting Wi-Fi and Bluetooth peripherals. The onboard computer is responsible for:

- (a) Establishing a serial connection between Raspberry Pi and Arduino over USB under ROS network, to send and receive ROS messages using roserial_python node.
- (b) Acquiring the depth images and points cloud from Xbox Kinect camera sensor and convert it to laser scans passing it through Wi-Fi to the command station using depth_image_to_laser_scan node.
- (c) Command the Arduino which is connected through USB by the commands received through Wi-Fi from the command station.

3. Micro-controller unit (Arduino Mega):

It is a micro-controller unit that is dedicated to the main DC motors with their attached magnetic encoders. Arduino Mega has a 16 MHz CPU frequency and 256 KB flash memory in addition to 16 analog input pins, 54 digital I/O pins 15 of them can be used for PWM (Pulse Width Modulation). It is equipped with an Adafruit v1 motor driver shield that supports up to 4 DC motors. It is responsible for

- (a) Sending the encoder counts status after processing it to the command station. It sends it to the Raspberry Pi over USB then to the command station over Raspberry Pi Wi-Fi, the encoder status is published over “/encoder” topic under the ROS network.
- (b) Execution of the motor commands sent by the command station over “cmd-vel” topic.

Results and discussion

Two different environments had been mapped by the robot. The first was a tall corridor with some obstacles while the second was a large hall with many obstacles. The two environments show successful identification and mapping of objects greater than 30 cm × 30 cm × 30 cm.

Corridor setup result

Figure 10 shows the real environment. On the right, an RGB photo of the corridor with different objects. On the left, the occupancy grid map is located where black lines are the obstacles, white spaces mean free of obstacle, and gray means no information whether occupied or free. The system had succeeded in capturing the walls and the obstacles as well.



Fig. 10 Corridor result



Fig. 11 Wide area

Wide area setup result

In Fig. 11, another wide setup is presented. The map had succeeded in capturing the three boxes in the middle in addition to the desks.

The two experiments show the functionality of the system and its ability to map and localize itself simultaneously.

Conclusions

This paper presents a low-cost simultaneous localization and mapping system for commercial applications. It can explore its environment by mapping the obstacles as well as localizing itself using noisy sensors. Computer vision and stereo imaging algorithms are used to extract depth information out of Kinect low-cost stereo camera to help in mapping the environment instead of using high-cost Laser scanners (Lidars). The Rao-Blackwellized particle filter, based on the Bayes rule, worked in filtering the sensor noise and in performing the simultaneous localization and mapping (SLAM) effectively. The Raspberry Pi board is the main onboard computer used to communicate wirelessly with the commanding station, while the Arduino board was dedicated to motors and their encoders. ROS powered the robot and made its programming more modular and maintainable, as well as upgradeable and integrable with further future work. This SLAM robotic system provides a cheap mature platform for developing and testing further autonomous navigation components such as path planning and following.

Abbreviations

CML	Concurrent localization and mapping
GPS	Global Positioning System
IPS	Indoor Positioning System
ROS	Robot Operating System
SLAM	Simultaneous localization and mapping

Acknowledgements

Not applicable.

Authors' contributions

This work is a part of a master thesis by Eng. Muhamamd Asem (MA) under the supervision of Professor Ayman Hamdy Kassem (AK). AK was responsible about the idea of the paper and the sequence of ideas and the accuracy of the results and was a major contributor in writing the manuscript. MA was responsible about the software and hardware implementation of the SLAM system. All authors have read and approved the manuscript.

Authors' information

The first author is Professor Ayman Hamdy Kassem a professor of Dynamics and Control at Aerospace Engineering Department, Cairo University. Email: akassem@cu.edu.eg.

The second author Eng. Muhammad Asem is a Master student from Aerospace Engineering Department.

Funding

Not applicable.

Availability of data and materials

Not applicable: the word "data" if shown in the text means the internal sensors data which cannot be reproduced/or used outside our environment to be useful for other researchers.

Declarations**Consent for publication**

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 27 February 2023 Accepted: 30 June 2024

Published online: 13 July 2024

References

1. Cadena C, Carrilho L, Latif Y, Scaramuzza D, Neira J, Reid I, Leonard JJ (2016) Past, present, and future of simultaneous localization and mapping: towards the robust-perception age. *IEEE Trans Rob* 32(6):1309–1332
2. Sebastian Thrun, Wolfram Burgard and Dieter Fox (2005) Probabilistic robotics. MIT press, ISBN: 9780262201629
3. Castellanos JA, Tardos JD, Schmidt G (1997) Building a global map of the environment of a mobile robot: the importance of correlations. *Proc Int Conf Robot Autom* 2:1053–1059. <https://doi.org/10.1109/ROBOT.1997.614274>
4. Hahnel D, Burgard W, Fox D, Thrun S (2003) An efficient fastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)* (Cat. No.03CH37453), vol 1. pp 206–211. <https://doi.org/10.1109/IROS.2003.1250629>
5. Chatila R, Laumond J (1985) Position referencing and consistent world modeling for mobile robots. *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. pp 138–145. <https://doi.org/10.1109/ROBOT.1985.1087373>
6. Smith R, Self M, Cheeseman P (1990) Autonomous robot vehicles. Springer Verlag New York, Inc, New York, pp 167–193
7. Crowley JL (1989) World modeling and position estimation for a mobile robot using ultrasonic ranging. *Proceedings, 1989 International Conference on Robotics and Automation*, vol 2. pp 674–680. <https://doi.org/10.1109/ROBOT.1989.100062>
8. Durrant-Whyte HF (1988) Uncertain geometry in robotics. *IEEE J Robot Autom* 4(1):23–31
9. Leonard JJ, Durrant-Whyte HF (1991) Simultaneous map building and localization for an autonomous mobile robot. *Proceedings IROS '91: IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*, vol 3. pp 1442–1447. <https://doi.org/10.1109/IROS.1991.174711>
10. Mohanty S, Naskar AK (2019) Analysis of the Performance of Extended Kalman Filtering in SLAM Problem. 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), Paris, France. pp 1031–1036. <https://doi.org/10.1109/CoDIT.2019.8820404>
11. Yatim NM, Buniyamin N (2015) Particle filter in Simultaneous Localization And Mapping (Slam) using differential drive mobile robot. *J Teknol* 77(20). <https://doi.org/10.11113/jt.v77.6557>
12. Grisetti G, Stachniss C, Burgard W (2007) Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Trans Rob* 23(1):34–46. <https://doi.org/10.1109/TRO.2006.889486>
13. Dellaert F, Kaess M (2006) Square Root SAM: simultaneous localization and mapping via square root information smoothing. *Int J Robot Res* 25(12):1181–1203. <https://doi.org/10.1177/0278364906072768>
14. Feng Lu, Millios E (1997) Globally consistent range scan alignment for environment mapping. *Auton Robot* 4(4):333–349
15. Olson E, Leonard J, Teller S (2006) Fast iterative alignment of pose graphs with poor initial estimates. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. pp 2262–2269. <https://doi.org/10.1109/ROBOT.2006.1642040>
16. Moral, Pierre Del and A. Doucet. "Particle methods: An introduction with applications." *Esaim: Proceedings* 44 (2014): 1–46.
17. Leonard JJ, Durrant-Whyte HF (1991) Mobile robot localization by tracking geometric beacons. *IEEE Trans Robot Autom* 7(3):376–382. <https://doi.org/10.1109/70.88147>

18. Zhang G, Suh IH (2009) Mathematical modeling of the prediction mechanism of sensory processing in the context of a Bayes filter. 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp 3937–3942. <https://doi.org/10.1109/IROS.2009.5353957>
19. Smith R, Self M, Cheeseman P (1990) Estimating Uncertain Spatial Relationships in Robotics. In: Cox JJ, Wilfong GT (eds) *Autonomous Robot Vehicles*. Springer, New York. https://doi.org/10.1007/978-1-4613-8997-2_14
20. Thrun S, Montemerlo M, Koller D, Wegbreit B, Nieto J, Nebot E (2004) Fastslam: An efficient solution to the simultaneous localization and mapping problem with unknown data association. *J Mach Learn Res* 4(3):380–407
21. Chen W et al (2022) SLAM overview: from single sensor to heterogeneous fusion. *Remote Sens* 14:6033. <https://doi.org/10.3390/rs14236033>
22. Zhang F, Li S, Yuan S, Sun E, Zhao L (2017) Algorithms analysis of mobile robot SLAM based on Kalman and particle filter. 2017 9th International Conference on Modelling, Identification and Control (ICMIC). pp 1050–1055. <https://doi.org/10.1109/ICMIC.2017.8321612>
23. Murphy K, Russell S (2001) Rao-blackwellised particle filtering for dynamic Bayesian networks. *Sequential Monte Carlo methods in practice*. Springer New York, New York, pp 499–515
24. Kuptamete C, Aunsri N (2022) A review of resampling techniques in particle filtering framework. *Measurement*. 193:110836
25. Xu X, Pang F, Ran Y, Bai Y, Zhang L, Tan Z, Wei C, Luo M (2021) An indoor mobile robot positioning algorithm based on adaptive federated Kalman filter. *IEEE Sens J* 21(20):23098–107
26. Elaraby AF, Hamdy A, Rehan M (2018) A Kinect-based 3D object detection and recognition system with enhanced depth estimation algorithm. 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada. pp 247–252. <https://doi.org/10.1109/IEMCON.2018.8615020>
27. Kassem AH (2015) Rover trajectory planning via simulation using incremented particle swarm optimization. *Proceedings of the 2015 International Conference of Scientific Computing, CSC 2015, Las Vegas, Nevada, USA, July 27–30*. pp 140–146
28. Jardine PT, Kogan M, Givigi SN, Yousefi S (2019) Adaptive predictive control of a differential drive robot tuned with reinforcement learning. *Int J Adapt Control Signal Process* 33:410–423
29. Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*
30. Yang Q, Wang L, Ahuja N (2010) A constant-space belief propagation algorithm for stereo matching. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp 1458–1465
31. Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J et al (2009) ROS: an open-source Robot Operating System. *ICRA workshop on open source software*, vol 3, No. 3.2. p 5
32. Koubaa A (2016) *Robot Operating System (ROS) The Complete Reference (Volume 1)*. Springer Nature (<https://link.springer.com/book/10.1007/978-3-319-26054-9>)

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.