A Simple Integrative Solution For Simultaneous Localization And Mapping

Min Raj Nepali *nminraj92@gmail.com*
Research Associate
Nitte Meenakshi Institute of Technology
Bangalore, 560064, India

Dubey Aditya Housila Prasad *adityad031@gmail.com*
Research Associate
Nitte Meenakshi Institute of Technology
Bangalore, 560064, India

Susheel Balasubramaniam *susheels88@gmail.com*
Research Associate
Nitte Meenakshi Institute of Technology
Bangalore, 560064, India

Venkatesh EN *venkateshen1@gmail.com*
Research Associate
Nitte Meenakshi Institute of Technology
Bangalore, 560064, India

Ashutosh *ashu.studsat@gmail.com*
Research Associate
Nitte Meenakshi Institute of Technology
Bangalore, 560064, India

Abstract

Simultaneous Localization and Mapping is a method used to find the location of a mobile robot while at the same time build a constructive map of its surrounding environment. This paper gives a brief description about a simple integrative SLAM technique using a Laser Range Finder (LRF) and Odometry data, primarily for indoor environments. In this project, a solution for the SLAM problem was implemented on a differential drive mobile robot equipped with a SICK laser scanner.

Keywords: Localization, LRF, Feature Based Mapping, Odometry, SLAM, Split and Merge.

1. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a method used by mobile robot placed in an unknown location in an unknown environment to incrementally build a consistent map of this environment while simultaneously keeping the track of current location within this map. An intuitive understanding of the SLAM process can be conveyed through following example: Consider a simple mobile robot: a set of wheels connected to a motor and a laser range finder (LRF). The LRF sends out series of beams at small regular intervals and obtains the distance of objects from it at a given angular resolution. This information is stored in a file that can be used to build the map.

Rather than keep track of a large number of points to build a map (termed as point-based mapping) we instead extract geometric features from the information obtained and keep track of those instead. This results in a significant reduction in computational complexity. The geometric features chosen in this implementation are lines, since they are present predominantly in indoor environments.

The SICK LMS-100 Laser range finder is used in this implementation. It has a 270° field view around the robot with angular step readings at 0.5°/0.25° resolution. The long range scan of about 20m makes it distinctive for its selection. Wheel encoders allow us to ascertain the orientation and speed of the robot. Motor drivers are used to control the speed of the wheels using a PID algorithm. The encoders from which odometry data is ascertained and the drivers are interfaced with a microcontroller.

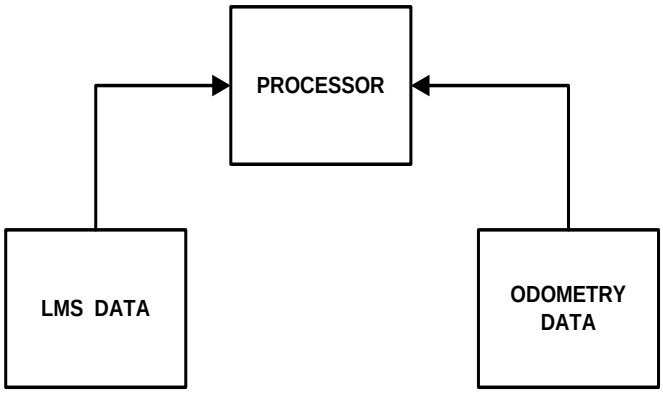

FIGURE 1: System setup.

The microcontroller transmits Odometry data to the computer for localization. The flowchart describing the whole process is shown below.

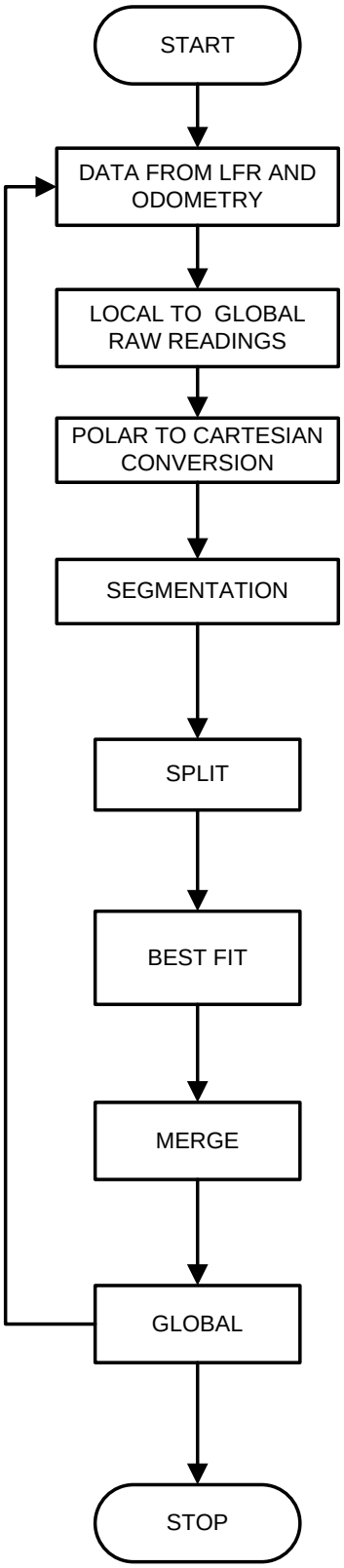

FIGURE 2: Flowchart of SLAM process.

2. ODOMETRIC LOCALIZATION

2.1. Representing Robot Position

The axes X and Y state an random inertial basis on the plane as the global reference frame or the origin.

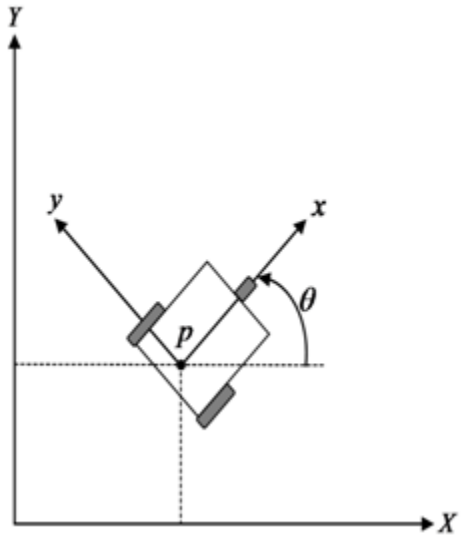

FIGURE 3: Global reference frame and the robot local reference frame.

To state the location of the robot, we select a point P on the robot frame as its position reference point. {x, y} outlines two axes relative to P on the robot frame and is thus the robot's local reference frame. The position of the robot P in the global reference frame is detailed by coordinates x and y, and the angular alteration between the global and local reference frames is given by θ. We can describe the pose of the robot as a vector with these three elements.

$$P_{xy} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad (1)$$

To describe robot motion in terms of component motions, it will be needed to map motion along the axes of the global reference frame to motion along the axes of the robot's local reference frame. This mapping is obtained using the orthogonal rotation matrix:

$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (2)$$

This matrix can be used to map motion in the global reference frame {x, y} to motion in terms of the local reference frame {x, y}. This operation is denoted by $R(\theta)\ P_{xy}$ because the computation of this operation depends on the value of θ:

$$P_{xy} = R(\theta)P_{xy} \qquad (3)$$

2.2. Forward Kinematic Model

The forward kinematics of the mobile robot captures how the robot moves, given its geometry and the speeds of its wheels. For example, the differential drive robot has two wheels, with radius r_r and r_l. Robot position reference point **P** cantered between the two drive wheels, and distance between the wheels ***l*.** Given r_r, r_l, l, θ and the spinning speed of each wheel, φ_r and φ_l, a forward

kinematic model would predict the robot's total speed in the global frame of reference. i.e $P_{xy}\ \{x, y, \theta\}$.

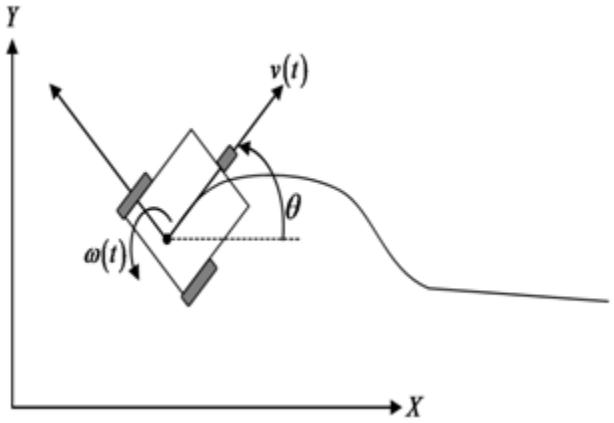

FIGURE 4: A differential-drive robot in its global reference frame.

From P_{xy} we know that we can compute the robot's movement in the global reference frame from motion in its local reference frame: $\dot{P}_{xy}=R(\theta)^{-1}\dot{P_{xy}}$.

Suppose that the robot's local reference frame is aligned such that the robot moves forward along$+x$, as shown in figure. First consider the contribution of each wheel's spinning speed to the translation speed at *P* in the direction of $+x$. If one wheel spins while the other wheel contributes nothing and is stationary, since *P* is halfway between the two wheels, it will move instantaneously with half the speed: $\dot{x}_r = (1/2)r_r\varphi_r$ and $\dot{x}_l = (1/2)r_l\varphi_l$.In a differential drive robot, these two contributions can simply be added to calculate the $\dot{x}$ component of $\dot{P_{xy}}$. Consider, for example, a differential robot in which each wheel spins with equal speed but in opposite directions. The result is a stationary, spinning robot. As expected, $\dot{x}$ will be zero in this case.. Neither wheel can contribute to sideways motion in the robot's reference frame, nor so $\dot{y}$ is always zero. Finally, we must compute the rotational component $\dot{\theta}$ of $\dot{P_{xy}}$. Once again, the contributions of each wheel can be computed independently and just added. Consider the right wheel (wheel 1). Forward spin of this wheel results in counter-clockwise rotation at point *P*. Recall that if the right wheel spins alone, the robot pivots around the left wheel. The rotation velocity ω_r at *P* can be computed because the wheel is instantaneously moving along the arc of a circle of radius *2l*:

$$\omega_r = \frac{r_r\varphi_r}{2l} \qquad (4)$$

The same calculation applies to the left wheel, with the exception that forward spin results in *clockwise* rotation at point *P*:

$$\omega_l = \frac{r_l\varphi_l}{2l} \qquad (5)$$

Combining these individual formulas yields a kinematic model for the differential-drive robot:

$$\dot{P_{xy}} = R(\theta)^{-1}\begin{bmatrix}\frac{r_r\,\varphi_r + r_l\varphi_l}{2}\\ 0\\ \frac{r_r\varphi_r - r_l\varphi_l}{2l}\end{bmatrix} \qquad (6)$$

3. MOTION MODEL

For a differential-drive robot the position can be estimated starting from a known position by integrating the movement (summing the incremental travel distances). For a discrete system with a fixed sampling interval Δt the incremental travel distances $(\Delta x, \Delta y, \Delta\theta)$ are:

$$\Delta x = \Delta s \cos\left(\theta + \frac{\Delta\theta}{2}\right) \quad (7)$$

$$\Delta y = \Delta s \sin\left(\theta + \frac{\Delta\theta}{2}\right) \quad (8)$$

$$\Delta\theta = \frac{\Delta s_r - \Delta s_l}{l} \quad (9)$$

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2} \quad (10)$$

Where,

$(\Delta x, \Delta y, \Delta\theta)$ = Path traveled in the last sampling interval.

$(\Delta s_x; \Delta s_x)$ = Travelled distances for the right and left wheel respectively.

Thus we get the updated position *P(k+1)*

$$p(k+1) = p(k) + \begin{bmatrix} \Delta s \cos\left(\theta + \frac{\Delta\theta}{2}\right) \\ \Delta s \cos\left(\theta + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta s \cos\left(\theta + \frac{\Delta\theta}{2}\right) \\ \Delta s \cos\left(\theta + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} \quad (11)$$

By using the relation for Δs and $\Delta\theta$, we further obtain the basic equation for odometric position update (for differential drive robots):

$$p(k+1) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{l}\right) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{l}\right) \\ \frac{\Delta s_r - \Delta s_l}{l} \end{bmatrix} \quad (12)$$

As we discussed earlier, odometric position updates can give only a very rough estimate of the actual position. Owing to integration errors of the uncertainties of $P(k)$ and the motion errors during the incremental motion$(\Delta s_r; \Delta s_l)$, the position error based on odometry integration grows with time.In the next step we will establish an error model for the integrated position $P(k+1)$ to obtain the covariance matrix of the odometric position estimate $P(k+1)$. To do so, we assume that at the starting point the initial covariance matrix $P(k)$ is known. For the motion increment $(\Delta s_r; \Delta s_l)$ we assume the following covariance matrix Q

$$Q = cov(\Delta s_r, \Delta s_r) = \begin{bmatrix} k_r |\Delta s_r| & 0 \\ 0 & k_l |\Delta s_l| \end{bmatrix} \quad (13)$$

Where k_r and k_l are error constants representing the nondeterministic parameters of the motor drive and the wheel-floor interaction. As you can see, in equation above we made the following assumptions: (i) The two errors of the individually driven wheels are independent; (ii) The variance of the errors (left and right wheels) is proportional to the absolute value of the travelled distances. These assumptions, while not perfect, are suitable and will thus be used for the further development of the error model. The *motion errors* are due to imprecise movement because of deformation of wheel, slippage, unequal floor, errors in encoders, and so on. The values for the

error constants k_r and k_l depend on the robot and the environment and should be experimentally established by performing and analysing representative movements.

If we assume that *P* and $q = (\Delta s_r; \Delta s_l)$ are uncorrelated and the derivation of *f* is reasonably approximated by the first-order Taylor expansion (linearization), we conclude, using the error propagation law:

$$P(k+1) = F_p(k).P(k).F_p{}^T(k) + F_q(k).Q.F_p{}^T(k) \quad (14)$$

$$F_p(k) = \begin{bmatrix} \frac{\partial p}{\partial x} & \frac{\partial p}{\partial y} & \frac{\partial p}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\Delta s\, sin\left(\theta + \frac{\Delta\theta}{2}\right) \\ 0 & 1 & \Delta s\, sin\left(\theta + \frac{\Delta\theta}{2}\right) \\ 0 & 0 & 1 \end{bmatrix} \quad (15)$$

$$F_q(k) = \begin{bmatrix} \frac{\partial p}{\partial \Delta s_r} & \frac{\partial p}{\partial \Delta s_l} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\left(\theta + \frac{\Delta\theta}{2}\right) - \frac{\Delta s}{2b}\sin\left(\theta + \frac{\Delta\theta}{2}\right) & \frac{1}{2}\cos\left(\theta + \frac{\Delta\theta}{2}\right) + \frac{\Delta s}{2b}\sin\left(\theta + \frac{\Delta\theta}{2}\right) \\ \frac{1}{2}\sin\left(\theta + \frac{\Delta\theta}{2}\right) + \frac{\Delta s}{2b}\cos\left(\theta + \frac{\Delta\theta}{2}\right) & \frac{1}{2}\sin\left(\theta + \frac{\Delta\theta}{2}\right) - \frac{\Delta s}{2b}\cos\left(\theta + \frac{\Delta\theta}{2}\right) \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \quad (16)$$

Once the error model has been established, the error parameters must be specified. One can compensate for deterministic errors properly calibrating the robot. However the error parameters specifying the nondeterministic errors can only be quantified by statistical (repetitive) measurements.

4. MAPPING

Mapping is the subject of integrating the readings gathered with the robot's sensors into a given representation. Principal aspects in mapping are the representation of the surroundings and the interpretation of sensor data. While on the other hand, localization is the problem of guesstimating the pose of the robot relative to a map. Normally, one identifies between pose tracking, where the initial pose of the vehicle is well-known, and global localization, in which no a priori knowledge about the starting position is set.

4.1. Feature Based Mapping

The primary issue is how to accurately match sensed data against information in a priori map or information that has been collected so far. There are two common matching techniques that have been used in mobile robotics: point based matching and feature-based matching. Instead of working directly with raw scan points, feature based matching first transforms the raw scans into geometric features. These extracted features are used in the matching in the next step. This approach has been studied and employed intensively in recent research on robot localization, mapping, feature extraction, etc. Being more compact that they require much less storage and still provide rich and accurate information, algorithms based on parameterized geometric features are expected to be more efficient compared to point-based algorithms.

Among many geometric primitives, line segment is the simplest one. It is easy to describe most office environments using line segments. A paper by Nguyen et all [2005] comes to the conclusion that the split and merge and *Incremental* are the preferred candidates for SLAM, because of their speed and good correctness. They mention that for real-time applications, Split-and-Merge is clearly the best choice by its superior speed. It is also the first choice for localization problems with a priori map, where false pose is not very important.

4.2. Landmark Extraction

Landmarks are features which can easily be re-observed and distinguished from the environment. These are used by the robot to find out where it is (to localize itself). Landmarks should be re-

observable by allowing them for example to be viewed (detected) from different positions and thus from different angles.

Landmarks should be unique enough so that they can be easily identified from one time-step to another without mixing them up. In other words if you re-observe two landmarks at a later point in time it should be easy to determine which of the landmarks is which of the landmarks we have previously seen. If two landmarks are very close to each other this may be hard.

One can easily translate a line into a fixed point by taking another fixed point in the world coordinates and calculating the point on the line closest to this fixed point. Using the robots position and the position of this fixed point on the line it is trivial to calculate a range and bearing from this. Using simple trigonometry one can easily calculate this point.
Here illustrated using the origin as a fixed point:

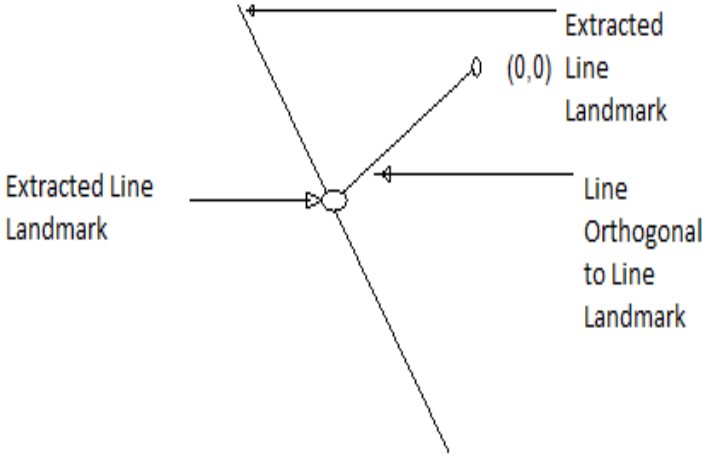

FIGURE 4: Getting extracted line landmark as a point.

4.3. Map Building

The sensor used to measure environment information and build environment map is a 2D laser scanner SICK**.** It has a scanning angle of 270 degrees with an angular resolution of 0.5 degree and a range error of less than 5mm. A local map is the representation of environment that laser range scanner perceives from its current position. A global map is the representation of environment that has been perceived previously. It is yielded by merging all local maps built previously.

4.4. Local Map Building

The formation of a local map consists of a number of steps and is implemented as follows:

i) Clustering of points into segments
From the first to the last point, if the distance between two neighbouring points is great than a threshold, then whole scan is split between these two points. After testing all distance of neighbouring points, a scan is split into different regions, denoted with Ri. Each region Ri consists of a set of points from Si to Et, where Si and Ei are respectively the first and the last point of the segment. Removal of Stray Segments: The number of points in each segment is counted. If the number of points in each segment is less than a given threshold, the segment is removed.

ii) The Split-and-Merge Algorithm
The algorithm splits the initial line by searching the particular point in between with the largest perpendicular line distance. This point is set as a new vertex, respectively a new segment endpoint, and builds two new segments with the former given endpoints. For the thereby new

generated segments the algorithm searches again the vertices. The algorithm stops the recursion in case that the distance to the farthest found point lies below a predefined threshold. The initial line is specified by the first and the last captured scan point. These therefore build the first and respectively the last vertex of our resulting Polyline.

The 'Merging' component of the Split-and-Merge Algorithm involves calculating the distance between the two adjacent points of each vertex derived from the Split procedure. If the distance is found to be greater than a given threshold, the vertex is considered a stray point, and is ignored.

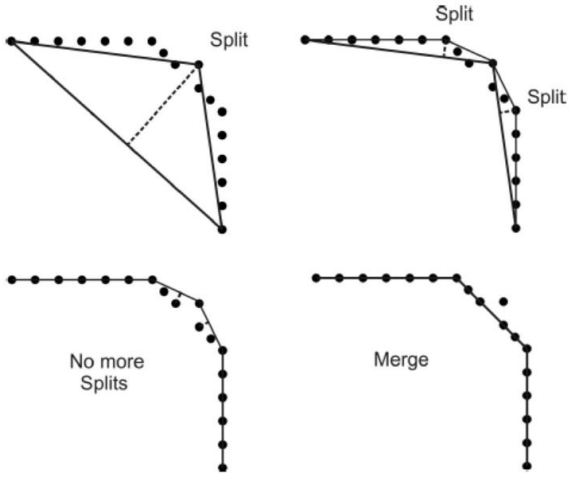

FIGURE 5: Split and Merge (Iterative end point fit).

iii) Finding the best-fit line using the method of least squares
Having obtained the points using the split-and –merge algorithm that correspond to the vertices in each segment, the best fit line in the least-squared sense is obtained, considering all the points in each segment derived.

The basic problem is to find the best fit straight line $y = ax + b$ given that, for $n \in \{1,\ldots Ng\}$, the pairs (xn, yn) are observed. The method easily generalizes to finding the best fit of the form

$$y = a_1 f_1(x) + \cdots + c_k f_k(x) \quad (17)$$

It is not necessary for the functions f_K to be linearly in x – all that is needed is that y is to be a linear combination of these functions. It can be shown that the expression for the landmark extracted (r and α) from the feature line is as follows:

$$\alpha = \frac{1}{2} atna2 \left\{ \frac{-2\sum(\bar{y}-y_i)(\bar{x}-x_i)}{\sum(\bar{y}-y_i)^2 - (\bar{x}-x_i)^2} \right\} \quad (18)$$

$$r = \bar{x}\cos(\alpha) + \bar{y}\sin(\alpha) \quad (19)$$

$$\bar{x} = \frac{1}{n}.\sum x_i \qquad \bar{y} = \frac{1}{n}.\sum y_i \quad (20)$$

Where

r, α - range and bearing of the extracted feature point
x, y- x and y co-ordinates of all points in the line segment

iv) Finding projections of the line segments on the best-fit line
Having found out the parameters of the best fit line, which also are the feature points extracted, we find the projection of the vertices on the best fit line in the following manner:

$$m = -\frac{cos\alpha}{sin\alpha} = -sec\alpha \quad (21)$$

$$c = \frac{R}{sin\alpha} \quad (22)$$

$$y_s = m y_p + m x_p + \frac{c}{m^2+1} \quad (23)$$

$$x_s = m y_p + x_p - \frac{mc}{m^2+1} \quad (24)$$

But since if α = 0,180 or -180 we obtain an exception, we use the following formula in these cases:

$$X_{s/e} = X_{start/end} \quad (25)$$

$$Y_{s/e} = Y_{start/end} \quad (26)$$

4.5. Global map building

The main problem of global map updating is to solve the correspondence problem between l i e segments respectively from local map and global map. If a line segment in local map has corresponding line segment in current global map, then they are merged into one to update this line segment in current global map. Those line segments in local map that have not corresponding line segment in current global map are inserted into current global map according to their ordinal relation. The global map is built by merging repeatedly local maps. The ordinal relations of line segments are retained in merging.

To determine whether given pair of lines is sufficiently similar to warrant merging, we apply a merge criterion based on the chi-squared test. The coordinates and covariance matrices of the two lines as found by our line fitting algorithm are first represented with respect to a common pose i. We then apply the chi-squared test to determine if the difference between two lines is within the 3 sigma deviance threshold defined by the combined uncertainties of the lines.

The merge criterion is:

$$x^2 = (\delta L)^T (P_{L1}^i + P_{L2}^i)^{-1} \delta L < 3 \quad (27)$$

Where

P^i_{L1}, P^i_{L2} - corresponds to the covariance matrices of the landmark in the local and global frames respectively. The difference between the landmark co-ordinates in the local and global frames are given as follows:

$$\delta L = \begin{bmatrix} R_1^i - R_2^i \\ \alpha_1^i - \alpha_2^i \end{bmatrix} \quad (28)$$

If the chi-squared test is satisfied, the local and global co-ordinates are merged to obtain a single landmark.

The final merged landmark co-ordinates and covariance matrix is given by:

$$L_m^i = P_{L_m}^i\left(\left(P_{L_1}^i\right)^{-1}L_1^i + \left(P_{L_2}^i\right)^{-1}L_2^i\right) \quad (29)$$

$$P_{L_m}^i = \left(\left(P_{L_1}^i\right)^{-1} + \left(P_{L_2}^i\right)^{-1}\right)^{-1} \quad (30)$$

5.RESULTS AND ANALYSIS

The mobile robot was placed in a corridor. A wall following algorithm was implemented such that the robot followed the wall observed to the right. Fig (7) illustrates the odometry path followed by the robot following the wall and Fig (8) shows the map obtained, until the robot veered dangerously close to a staircase.

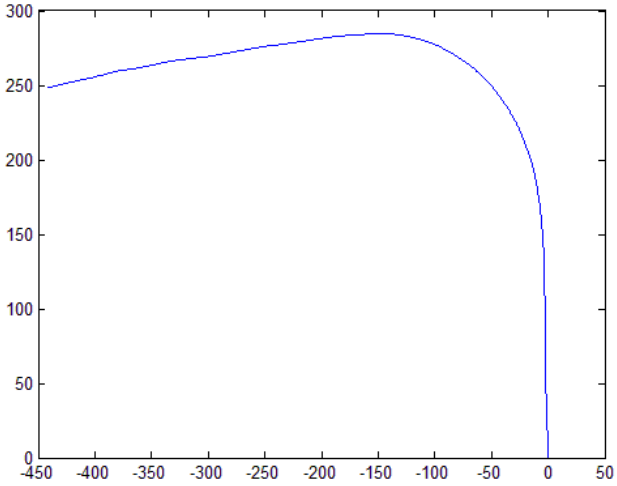

FIGURE 6: Localization data from odometry.

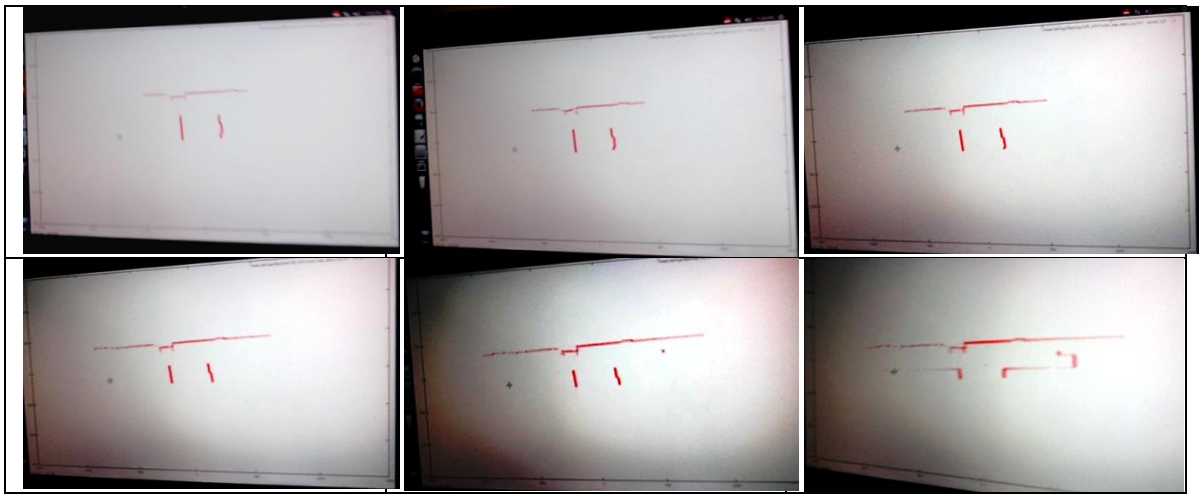

FIGURE 7: Plot of the map obtained after Global Mapping.

6. CONCLUSION

This paper illustrates a simplistic SLAM solution along with its implementation and results. The split-and-merge line feature and landmark extraction algorithms are explained in detail and are found to be accurate. The global map generated is found to be consistent in small-scale environments.

It is possible to refine the solution provided in this paper by implementing extended Kalman filters, in order to obtain the best estimate of the robot position and map points. Different algorithms that consider the SLAM process a non-linear process such as FAST-SLAM may be implemented as well.This current technology can be used in various applications such as aerospace, a mobile robot on the moon, a coal mine where human entry is difficult, underwater monitoring in small and difficult to reach places.

7.REFERENCES

[1] H. F. Durrant-Whyte, "Where am I? A Tutorial on Mobile Vehicle Localization," Industrial Robot, vol. 21, no. 2, pp. 11±16, 1994.

[2] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. "FastSLAM: A factored solution to the simultaneous localization and mapping problem". AAAI National Conference on Artificial Intelligence, pages 593–598, 2002.

[3] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, 2005.

[4] Lina M. Paz, Pedro Pini´es, Juan D. Tard´os, and Jos´eNeira, Large- Scale 6-DOF SLAM With Stereo-in-Hand, IEEE Transactions On Robotics, Vol. 24, No. 5, October 2008

[5] J.A. Castellanos and J.D. Tard´os. "Mobile Robot Localization and Map Building: A Multisensor Fusion Approach". Kluwer Academic Publishers, Boston, MA, 2000.

[6] W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. Sonarbased mapping with mobile robots using EM. In Proc. 16th International Conf. on Machine Learning, pages 67–76. Morgan Kaufmann, San Francisco, CA, 1999.

[7] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots," Machine Learning 31, 29–53 and Autonomous Robots 5, 253–271,(joint issue), 1998.

[8] Jose Guivant, Eduardo Nebot and Stephan Baiker. "Autonomous Navigation and Map building Using Laser Range Sensors in Outdoor Applications". Journal of Robotic Systems, Volume 17, No 10, October 2000

[9] Jaun D. Tardos , Jose Neira , Paul M. Newman , John J. Leonard "Robust Mapping and Localization in Indoor Environments using Sonar Data." The International Journal of Robotics Research Vol. 21, No. 4, April 2002.