Cumulative Probability based Surveying and Absolute Positioning System for Mobile Robots

Vaibhav Kulkarni [1], Mahadev Shirwaikar [2], Chetan Desai[3], Shambhavi Narvekar[4]– Member IEEE

[1] Masters Student (Embedded Systems), Department of Mathematics and Computer Science, TU-Eindhoven
[2] Masters Student, Department of Electrical Engineering, Indian Institute of Technology, Bombay
[3] Faculty, Department of Electronics & Telecommunication Engineering, Goa College of Engineering, Goa
[4] Masters Student, National Institute of Industrial Engineering, Mumbai

***Abstract* – Surveying is an essential part of any robot based surveillance application. Existing surveying algorithms have optimized shortest path and bend number of the path. We propose a novel probabilistic surveying algorithm and a metric viz. cumulative probability, to survey and evaluate the performance of path planning surveying algorithms. The proposed algorithm exhibits better performance with regards to area covered and the time taken, as compared to the traditional surveying algorithms such as flood fill and lawn mower for a vast majority of the cases. This paper will focus on a coverage path planning algorithm for mobile robots constrained to operate in the plane. For surveying applications that demand rapid deployment and quick synergy, precise knowledge of position is also of vital importance. In order to estimate robot's position while surveying with high precision, we propose an odometry based coordinate mapping strategy to assist the surveying application. It can be efficiently applied to a robot whose physical dimensions and mechanical specifications are known a priori. Using the theoretical models and practical measurements, we prove that our algorithm minimizes the time taken while surveying and object detection, in addition it provides higher accuracy in coordinate computation as compared to existing odometry methods employed while surveying.**

***Index Terms* — Coordinate mapping; Path planning; Probabilistic surveying algorithm; Shaft encoder; Wheel Odometry:**

I. INTRODUCTION

The surveying area problem is a research topic in the field of geographic information science and computer science. Motion planning algorithms [1] originally considered the start-goal problem whose solution determines a path (or trajectory) between two points [2]. The article presents a new topic in path planning for mobile robots which involves a sweeping operation [3] to fill a whole region with random obstacle detection. Surveying techniques play a great role in determining how quickly selective search for the target object in the least possible amount of time can be done. Algorithms for such classical surveying problems are abundantly available in literature viz. potential function approaches [4], [5] to a provably complete sensor-based method [6]. Nevertheless there is always a need to revisit these problems considering the relevant applications. We propose a new metric for evaluating the performance of such algorithms. We have attempted to modify the existing surveying algorithms to allow coverage of maximum sweep area and minimum time for object detection.

For any application that involves mobile objects, positioning is one of the most important aspect that needs to be taken into consideration. This need gains more importance in surveying applications to do path tracing and co-ordinate mapping. Position estimation can be done by using a technique known as odometry. An efficient algorithm for odometry should be capable of locating an object with high accuracy and precision at all times.

We have put forth an algorithm to enable reliable odometry on mobile robots for surveying tasks. This algorithm translates the location of a robot onto a universal coordinate mapping system. It requires the knowledge about the kinematics of the robot. The wheels of the robot are powered by DC motors and shaft encoders are used to provide the tracking of movement. The data obtained from the shaft encoders is used by the algorithm to compute the 2-D coordinates and the angle of orientation in a cumulative manner.

In section II, we have reviewed the existing methods of surveying and coordinate mapping. Section III has been used to explain the detailed design considerations required for the implementation the surveying algorithm and the odometry based coordinate mapping system. Section IV offers the observations and practical comparison between existing surveying and positioning algorithms and our proposed algorithm based on the UMBenchmark test [7]. We discuss the results and additional insights in section V and conclude our paper in section VI.

II. LITERATURE SURVEY

This section presents an extensive review of the existing surveying algorithms. The algorithms commonly used for object tracking are Lawn Mower and Flood which are presented below.

A. Lawn Mower

One of the most commonly used algorithms in robotics is the lawn mower algorithm. It involves a number of right angle turns proportional to the amount of area to be swept.

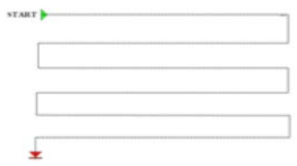

Figure 1. Lawn mower path

The robot moves in horizontal as well as vertical directions in alternate manner as per the need of the user. This algorithm ensures that entire area is covered [8].

B. Flood Fill

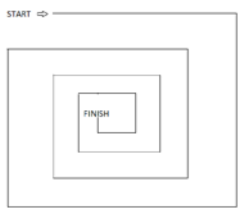

Figure 2. Flood fill path

Flood fill, also called seed fill, is an algorithm that determines the area connected to a given node in a multi-dimensional array. It is used for applications that involve quick survey of the area under test. There is a start node and an end node and the algorithm sweeps the two opposite ends of the area alternately while shrinking to the center gradually.

Many path planning algorithms and some coverage approaches assume that the robot knows the layout of the environment prior to the planning event. In many situations, this assumption may be unrealistic. Instead, the robot must use its on-board sensors to acquire information about the environment and perform coverage on-line. We use this sensor based coverage to explore and map the area.

C. Heuristic and randomized approaches

One class of heuristic algorithms for coverage employs an approach in which the robot is equipped with a simple set of behaviors (e.g. following a wall). A hierarchy of cooperating behaviors forms more complicated actions, such as exploration [9].

D. Cellular decompositions

Cellular decomposition can be in turn classified in to Approximate cellular decompositions [10], Semi-approximate [11] [12], Exact cellular decompositions [13], Boustrophedon decomposition [14], Optimal decomposition [15], Degenerate decomposition [16].

E. Positioning System

Positioning system can be relative or absolute. Multi robot systems employing relative positioning method allow the robots to find their location only with respect to each other. Thus it proves useful only in cases where there is a system formed due to multiple individual units. In absolute positioning, the robots are located with respect to a universal coordinate system. Global Positioning System (GPS) and differential GPS are the conventional methods of finding absolute positions that are used worldwide. The relative position between two individuals can be obtained by using the difference in their coordinates.

One of the most widely used odometry method for mobile robots makes use of the optical mouse sensors [17]. In this method, data is taken from the two optical sensors placed on the robot to compute the x, y coordinates [18]. The optical mouse sensors do not work well on transparent or reflective surfaces such as glass [19]. Any unevenness on the surface will also contribute to errors. Dead reckoning [20] (also called as deduced reckoning) is the process of calculating one's current position by using a previously determined position. However, it does not consider the turns that are taken while the robot is stationary. Another method for odometry is visual odometry which employs cameras and image processing. The precision of visual odometry system depends on the resolution of the images captured by the camera [21]. Recent research in odometry is focused towards visual odometry but the results are not sufficiently accurate.

III. IMPLEMENTATION

Existing surveying algorithms have optimized *shortest* path and *bend number* of the path [22]. We have explored another metric viz. cumulative probability, for measuring the performance of surveying algorithms. Cumulative probability refers to the probability that the value of a random variable falls within a specified range. Placement of the object is a random variable. Hence cumulative probability is a better metric for performance comparison of surveying algorithms.

A. Probabilistic Surveying Algorithm

This algorithm is based on the principle of cumulative probability. The path traversed by the robot while performing this surveying operation is as shown in the figure 3. This algorithm ensures that the robots survey the entire area in such a way that minimum time is required to locate an object while maximizing the cumulative probability of object detection at each instant of time.

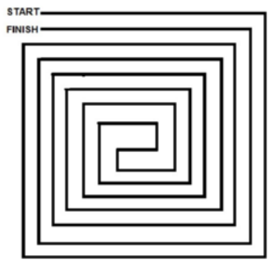

Figure 3. Probabilistic surveying algorithm path

Since we have proposed a new algorithm it is necessary to establish its credibility. This is achieved by making a comparison with standard surveying models. In order to reduce the complexity of the problem under consideration, we discretize the area in to a grid pattern. The surveying area is considered as a grid of dimensions 'A*B' with cell size 'S*S'.

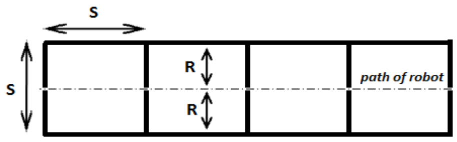

Figure 4. Area discretization into grids

a. Robot Initial Position

The robot is assumed to be initially placed at the center of a cell '1' as shown in the figure 4 and moves along the center line with the grid lines serving as the boundaries. The robot is equipped with a proximity sensor on each side having a range 'R' such that R=S/2. The robot dimensions are neglected for mathematical simplicity. A particular cell is termed explored when the robot has visited/traversed or lies in the range of the proximity sensors.

case(i): Object lies entirely in the cell.
case(ii): Object lies on the boundary (left) of cell.
case(iii): Object lies on the boundary (right) of cell.
case(iv): Object lies beyond the boundary (left) of cell.
case(v): Object lies beyond the boundary (right) of the cell.

When the robot travels along the straight line path as shown in the figure, it is able to detect the objects belonging to cases i, ii & iii. However the objects in cases iv & v are not detected as they lie beyond the range of the proximity sensors.

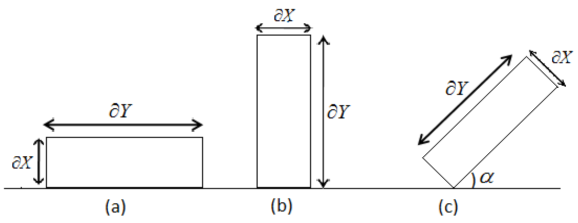

Figure 5. Object Location

Now, in cases iv & v, the object can lie beyond the boundary in different positions as shown in the figure 5. This contributes an increase in the area covered by the robot during its traversal. ∂z is the projection of the object on the plane perpendicular to robot's motion. In fig 5(a), the object lies along its length, ∂x along the boundary. Thus for this case, $\partial z=\partial x$. In fig 5(b), the object lies along its width, ∂y along the boundary. Thus for this case, $\partial z=\partial y$. In fig 5(c), the object lies at an angle $\alpha$with respect to the boundary. Thus using trigonometry, for this case, $\partial z=\partial x\cos\alpha$ or $\partial z=\partial y\cos\alpha$ depending on whether the length or width of the object subtends the angle $\alpha$with respect to the boundary.

When the robot traverses along a straight line path, it covers a strip of width R on each side. Hence the total area explored is 2R. The effective area explored depends on the object's location. Assuming that the adjacent strip paths have not been covered previously, the robot covers a strip of width $2\mathrm{R}+2\partial z$. However, if the adjacent strips have already been covered, the robot covers a width amounting to $2\mathrm{R}-2\partial z$. If one of the adjacent strips has been covered previously than the robot covers a strip of 2R.

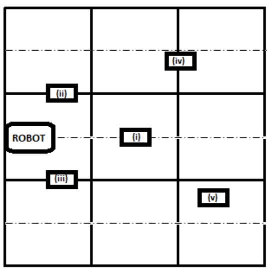

Figure 6. Area division in terms on strips

When robot is covering a strip of $2R$+$2\partial z$ it can cover case 1, 2, 3 of object position as shown in fig. 5 for 2r it is 1 & 2 or 2 & 3 and for $2R$-$2\partial z$ it can cover only 1. As can been seen $2R$+$2\partial z$ is the best option along which the robot can traverse followed by 2R and $2R$-$2\partial z$ respectively.

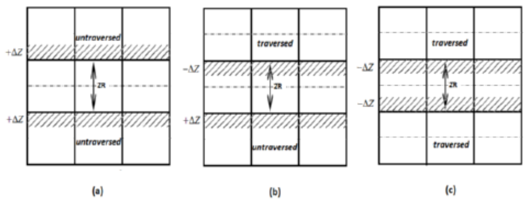

Figure 7. Area division while traversing

Using the above conditions we can test the efficiency of our algorithm with the standard algorithms- Lawn mower and Flood fill. The number of steps covered while surveying the whole area is the same in all three algorithms, i.e. $S^2 - 1$. Thus, the maximum time required to detect the object is the same in all the three cases. Since, we have expressed the area in terms of steps, the probability of an object lying in a cell of S*S is $\frac{1}{A\times B}$. There are a total of $A \times B$cells in which the object can lie. We find the probability of detecting the object for each of these cases sequentially. The probability of an object being found in the nth cell is given by

$$P_n = \frac{(S - \alpha \times \delta z)}{(A \times B - n)S - \sum_{i=0}^{n-1} \alpha_i \times \delta z}$$

Where is the size of each cell in cms. A and B are the dimensions of the area in terms of cells. α=0 when both the sides of i^{th} cell are uncovered.

α=1 when one side of the i^{th} cell is covered.
α=2 when both the sides of i^{th} cell are covered.

Based on the values of probability in the initial stages of cells, we claim that the probabilistic algorithm detects the object randomly placed in the area in the minimum time.

B. Odometry Based Coordinate Mapping

In this section, we have presented the details of the algorithm design. The mechanical specifications of the robot that are required are as follows:

1) Resolution of the shaft encoder used.
2) Width of the robot.
3) Circumference of the wheels.
4) Speed of the robot.

The incorporation of DC motors over stepper motors has been done to allow a tradeoff between cost of hardware and accuracy. The algorithm is applicable to a robot having a defined structural mechanism chosen by us due to lack of standardization. The robot has a two wheel drive mechanism with a castor wheel at the front. The coordinates of the robot are the coordinates of midpoint of the line joining the driving wheels of the robot i.e. C (x, y) as shown in fig. 8 which are computed after every ' T ' sec. estimation time.

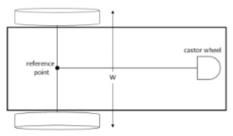

Figure 8. Robot chassis

a. Choice of reference point

Through our algorithm, we aim to develop a positioning system that will calculate the coordinates of the robot accurately. To achieve high resolution (mm scale), it is essential to choose a reference point on the robot in such a way that errors are reduced. Many points satisfy this requirement, most prominent among them being, center of the robot chassis, and center of wheel base or any of the wheels. We have chosen to use the center of the wheel base of the robot as the reference point for our algorithm for reasons of symmetry.

b. Robot movements

Now, in an estimation time interval 'T', the robot movement maybe classified as follows.
Movement 1: Straight line path with an inclination with respect to x-axis (-180 degrees < inclination < 180 degrees).

Movement 2: Turn with one motor driving the wheel and the other inhibited.

Movement 3: A combination of the straight line path and a turn.

Let $P1$ = number of pulses in time T given by the left shaft encoder and

$P2$ = number of pulses in time T given by the right shaft encoder, where T is the estimation time interval.

For calculating the distance covered by the robot in time T, the average of the number of pulses is taken.

$$P_{avg} = \frac{(P1 + P2)}{2} \quad (1)$$

Let 'D_w' is the diameter of the wheels in cm and 'D' be the distance covered in time T. The algorithm computes the coordinates of the robot after every estimation time T.

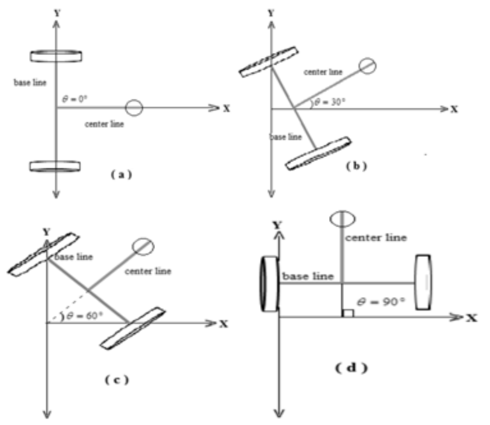

Figure 9. Schematic of a robot taking left turn for three consecutive T intervals

Now, the maximum distance covered by the robot in time T is

$$\frac{S \times C_w \times T}{60} \tag{2}$$

Where S is the speed of the robot in rpm and Cw is the circumference of the wheels in cm.

Similarly, the maximum angle $A\max$ covered by the robot in time T is calculated to be

$$\frac{S \times C_w \times T \times 360}{60 \times 2 \times \prod \times W} \text{ degrees} \tag{3}$$

Where W is the width of the robot in cm Simplifying the above equation, we get

$$A_{\max=} \frac{6 \times S \times D_w \times T}{W} \text{ degrees} \tag{4}$$

The maximum number of pulses obtained in time T from each of the shaft encoders is

$$P_{\max=} \frac{D_{\max}}{R} \tag{5}$$

Where R is the resolution of shaft encoders in cm/pulse and $D\max$ is the maximum distance covered by the robot in T.Therefore, distance covered by the robot in time T is given by

$$R \times P_{avg} cm \tag{6}$$

The difference of the no of pulses from shaft encoders is

$$P_{diff} = P_1 - P_2 \tag{7}$$

If $P_{diff} = 0$, it means the robot is moving along a straight line (robot movement 1).

If $P_{diff} = \pm P_{\max}$, it means the robot is taking a turn (robot movement 2).

If P_{diff} has an intermediate value, it means the robot has taken a turn for some time and moved in a straight line for the remaining estimation time. Positive value of P_{diff} indicates right turn and negative value indicates left turn.

When the robot is performing movement 1, $D = D_{\max}$

When the robot is performing movement 2, $A = A_{\max}$

C. Concept of angles

There are two different angles involved in the algorithm:

• Angle of Orientation

The angle at which the center line of the robot is oriented with respect to the X axis is called as the angle of orientation. The angle of orientation at time $k \times T$ is denoted as $\theta(T)$.

• Angle of Computation

The angle of computation is the angle that is used for computing the incremental coordinates ∂x and ∂y covered by the robot in time T. It is an imaginary angle that cannot be represented graphically. The angle of computation at time $k \times T$ is denoted as $\phi(T)$. we define a parameter Q which denotes the angle of orientation for movement 1 and the angle of computation for movements 2 and 3.

The angle of orientation is obtained as follows.

$$\theta(T) = \theta(T-1) + \Delta A(T, T-1) \tag{8}$$

The angle of computation is obtained as follows

$$\phi(T) = \phi(T-1) + [0.5 \times \Delta\theta(T, T-1)] + [0.5 \times \Delta\theta(T-1, T-2)] \tag{9}$$

Thus, the incremental distances in X and Y directions are

$$X = X + D \times \cos(Q) \tag{10}$$

$$Y = Y + D \times \sin(Q) \tag{11}$$

Therefore, we see that the incremental coordinates are computed after every T sec and are added to the coordinates computed in the previous T sec to obtain the present coordinates.

Flowchart for the proposed algorithm is presented in Figure 10.

D. Error calculation

The main source of error arises due to the fact that the robot is not aware of its position during the estimation time T. There is a difference between the following movements performed by the robot in an estimation time T.

1) Taking a turn for a time $x(x < T)$ and then moving straight for the remaining time $(T - x)$.

2) Going straight for a time $(T - x)$ and then taking a turn for the remaining time x.

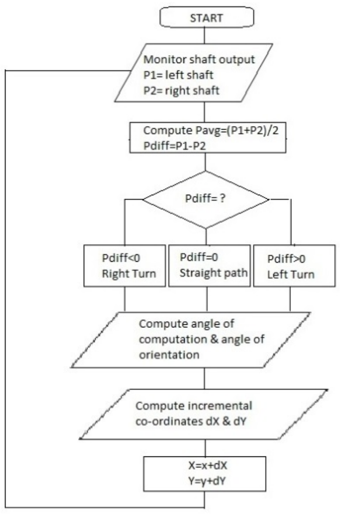

Figure 10. Flowchart for coordinate computation

We find the coordinates for the above two cases considering that the robot is placed at the origin.

The coordinates of the robot after time interval T for the following cases are:

Case 1:

$$X_1 = \left(\frac{S \times C_w \times x}{120}\right) \times \cos\left(\frac{3 \times S \times D_w \times x}{W}\right) + \left(\frac{S \times C_w \times (T - x)}{60}\right) \times \cos\left(\frac{6 \times S \times D_w \times x}{W}\right) \tag{12}$$

$$Y_1 = \left(\frac{S \times C_w \times x}{120}\right) \times \sin\left(\frac{3 \times S \times D_w \times x}{W}\right) + \left(\frac{S \times C_w \times (T - x)}{60}\right) \times \sin\left(\frac{6 \times S \times D_w \times x}{W}\right) \tag{13}$$

Case 2:

$$X_2 = \left(\frac{S \times C_w \times (T - x)}{60}\right) + \left(\frac{S \times C_w \times x}{120}\right) \times \cos\left(\frac{3 \times S \times D_w \times x}{W}\right) \tag{14}$$

$$Y_2 = \left(\frac{S \times C_w \times x}{120}\right) \times \sin\left(\frac{3 \times S \times D_w \times x}{W}\right) \tag{15}$$

The maximum error during every estimation time is

$$E_{\max} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{16}$$

These are not the only two cases that can occur during an estimation time. There are an infinite number of cases with different time intervals for turns and straight line paths but the two cases we considered above are the extremes. The maximum error can be reduced to half of its current value by always

E. Theoretical comparison

In this section, we have put forth a comparison between the readings obtained from the *Unidirectional Square-Path Test* [1] run conducted using the standard odometry algorithm proposed by Crowley et al [2] and our algorithm.

We performed theoretical calculations using the formulae put forth in both the algorithms. We considered that the robot has to traverse the four legs of the square path in the fourth quadrant of the Cartesian co-ordinate system. We have chosen to ignore the systematic and non-systematic errors so that the robot is assumed to reach back to the starting point.

Using the hardware available to us, the values of various parameters were found to be as follows: T =0.25 sec, S =60 rpm, W =15 cm, D_w =5 cm, R = 0.544 cm/pulse.

The readings corresponding to points when robot movement changes, obtained using the algorithm proposed by us, are as presented below:

Considering the midpoint of the two cases. So, we have,

$$E_{\max} = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{2} \tag{17}$$

The error reduces as the estimation time or speed of robot is decreased. An error in cm scale is observed with S < 100rpm and T < 1sec. This error is cumulative.

We can see that the Wheel Odometry Algorithm [2] works well when the robot is moving on a straight path but gives rise to errors when the robot encounters turns. A comparison of Table 1 and Table 2 shows that there is an error in the computation of x-coordinate and y-coordinate. However our algorithm (assuming there are no systematic and non-systematic errors) successfully provides the desired coordinates accurately.

IV. OBSERVATIONS AND PRACTICAL COMPARISONS

The robot should detect the object which is placed randomly in the area in minimum possible time. The ideal scenario is that the robot should find the object in the first few cells that it surveys. This can only happen if the robot traverses the area in such a way that the probability of finding the object is maximized. Different sets of experiments were conducted using areas of varying sizes. Only two tests have been presented here. One test was performed in an area of 5×6cells and the other was performed in 8×8 cells.

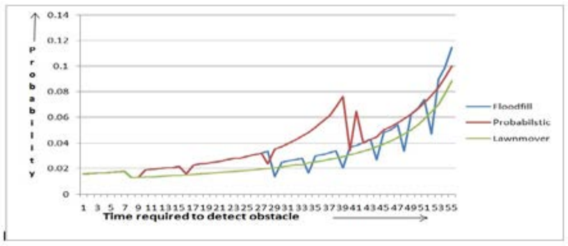

Figure 11. Trend of probability values for an area of 5*6 cells.

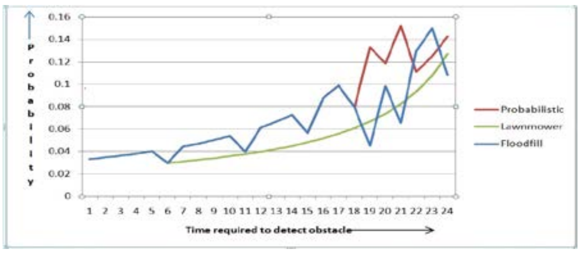

Figure 12. Trend of probability values for an area of 8*8 cells.

We performed the *Unidirectional Square-Path Test* run using the robot as per the mechanical specifications mentioned earlier. The practical readings, obtained using our proposed algorithm, and by Crowley's algorithm are as follows:

V. RESULTS AND DISCUSSIONS

The graphs show the probability curves for object detection v/s the time taken to detect the object. The values are very small in the first few cells that are traversed. This is because the sample space of probability is large in the initial stages of surveying. As the robot traverses through the area, the probability curves approach the value of unity. This indicates that the chance of finding the object in the cells increases successively as it traverses a greater distance. This is because the sample space decreases in the latter stages. Thus, higher the curve better is the surveying algorithm. As the size of the area increases, the probabilistic algorithm proves to be better.

Regarding the coordinate mapping algorithm, the theoretical comparison clearly suggests a significant error in Crowley's algorithm. This error is due to the contribution of preceding angle values on present co-ordinate values. Crowley et al [2] calculated the average orientation during a cycle by the previous estimate plus half the incremental change. As per our calculations, the average orientation during a cycle is given by the previous estimate plus half the incremental change in addition to half the previous incremental change as indicated by equations (9) and (10). This leads to a cumulative error which builds up for positive angles and reduces for negative angles. The practical run performed using UMBenchmark test shows the presence of systematic and non-systematic errors which masks the theoretical errors established above in Crowley's algorithm.

VI. CONCLUSION

An innovative algorithm for surveying was proposed. The performance metric of cumulative probability was introduced. The proposed algorithm exhibited better performance as compared to the traditional algorithms of flood fill and lawn mower for a vast majority of cases. An algorithm for wheel odometry has been developed. The coordinates have been computed by using wheels coupled with shaft encoders and driven by dc motors. The accuracy of the algorithm is proportional to the accuracy of measurement of robot kinematics. The accuracy of shaft encoder as stated in the datasheet is 0.544cm. If the wheel and encoder alignment is not precise, the accuracy of encoder is greatly reduced. Thus the surveying algorithm coupled with the coordinate mapping technique can be used to again higher accuracy, object detection in minimal time and faster deployment.

References:

[1]Shekhar, S.; Latombe, J.-C., "On goal recognizability in motion planning with uncertainty," *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on* , vol., no., pp.1728, 1733 vol. 2, 9-11 Apr 1991.

[2]Batalin, M.A.; Sukhatme, G., "The Design and Analysis of an Efficient Local Algorithm for Coverage and Exploration Based on Sensor Network Deployment," *Robotics, IEEE Transactions on* , vol.23, no.4, pp.661,675, Aug. 2007.

[3]J. VanderHeide and N.S.V. Rao, "Terrain coverage of an unknown room by an autonomous mobile robot", Technical report ORNL / TM-13117, Oak Ridge National Laboratory, Oak Ridge, TN (1995).

[4]D. Kurabayashi, J. Ota, T. Arai and E. Yoshida,

"Cooperative sweeping by multiple mobile robots", in: *Int. Conf. on Robotics and Automation* (1996).

[5]O. Khatib et al. "Real-time Obstacle avoidance for manipulators and mobile robots", Internat. J. Robotics Res. 5 (1986) 90–98.

[6]E. Rimon and D.E. Koditschek, "Exact robot navigation using artificial potential functions", IEEE Trans. Robotics Autom. 8(5) (1992) 501–518.

[7]Johann Borenstein, Liqiang Feng; "UMBmark: A benchmark test for measuring odometry errors in mobile robots"; 1995 SPIE Conference on Mobile Robots, Philadelphia.

[8]Alami, R.; Simeon, T., "Planning robust motion strategies for a mobile robot," Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on , vol., no., pp.1312,1318 vol.2, 8-13 May.

[9]Mare, J., "Path following algorithm for minimally specified lawn-mower type AUV missions," *OCEANS 2010 IEEE - Sydney* , vol., no., pp.1,5, 24-27 May 2010.

[10]Y.Y. Huang, Z.L. Cao and E.L. Hall, "Region filling operations for mobile robot using computer graphics", in: *Proceedings of the IEEE Conference on Robotics and Automation* (1986) pp. 1607–1614.

[11]Elfes A.,"Sonar-based real-world mapping and navigation," *Robotics and Automation, IEEE Journal of*, vol.3, no.3, pp.249,265, June 1987.

[12]H. Moravec and A. Elfes, "High resolution maps for wide angles sonar", in: *IEEE Int. Conf. on Robotics and Automation* (1985).

[13]S. Hert, S. Tiwari and V. Lumelsky, A terrain-covering algorithm for an AUV", Autonom. Robots 3 (1996) 91–119.

[14]V. Lumelsky, S. Mukhopadhyay and K. Sun, "Dynamic path planning in sensor-based terrain acquisition", IEEE Trans. Robotics Autom. 6(4) (1990) 462–472.

[15]F.P. Preparata and M.I. Shamos, "*Computational Geometry: An Introduction"* Springer-Verlag, Berlin, 1985 pp. 198–257.

[16]H. Choset, E. Acar, A. Rizzi and J. Luntz, "Exact cellular decompositions in terms of critical points of morse functions", in: *IEEE International Conference on Robotics and Automation*, San Francisco, CA (2000).

[17]Oliver Maya, Jan Schaeffner, Michael Maaser; "An Optical Window Positioning System for the Mass Market"; IHP GmbH, Frankfurt (Oder), Germany, 3rd workshop on Positioning, Navigation and Communication (WPNC"06) .

[18]Steven Bell;" High-Precision Robot Odometry Using an Array of Optical Mice"; Oklahoma Christian University.

[19]K.Nagatani S.Tachibana M.Sofue Y.Tanaka;" Improvement of Odometry for Omnidirectional Vehicle using Optical Flow Information"; Okayama University.

[20] George J. Geier, Ardalan Heshmati, Kelly G Johnson, Patricia McLain; "Position and velocity estimation system for adaptive weighing of GPS and dead reckoning information". US 5416712 A, 1993.

[21]Atsushi Sakai, Yuya Tamura, Yoji Kuroda; "Visual odometry using feature point and ground plane environment"; ISR / ROBOTIK 2010.

[22]W. Huang, Optimal line-sweep decompositions for coverage algorithms, Technical Report 00-3, Rensselaer Polytechnic Institute, Department of Computer Science, Troy, NY (2000).

[23]Crowley, J.L. and Reigner, P., 1992, Asynchronous Control of Rotation and Translation for a Robot Vehicle." Robotics and Autonomous Systems, Vol 10, 1992.