A Path Planning Technique For Autonomous Mobile Robot Using Free-Configuration Eigenspaces

Shyba Zaheer *s.shyba@gmail.com*
Department of Electrical & Electronics Engineering
T.K.M. College of Engineering
Kerala, India

Tauseef Gulrez *gtauseef@ieee.org*
Virtual and Simulations of Reality (ViSOR) Lab,
Department of Computing, Macqaurie University
2109 NSW, Sydney, Australia.

Abstract

This paper presents the implementation of a novel technique for sensor based path planning of autonomous mobile robots. The proposed method is based on finding free-configuration eigen spaces (FCE) in the robot actuation area. Using the FCE technique to find optimal paths for autonomous mobile robots, the underlying hypothesis is that in the low-dimensional manifolds of laser scanning data, there lies an eigenvector which corresponds to the free-configuration space of the higher order geometric representation of the environment. The vectorial combination of all these eigenvectors at discrete time scan frames manifests a trajectory, whose sum can be treated as a robot path or trajectory. The proposed algorithm was tested on two different test bed data, real data obtained from Navlab SLAMMOT and data obtained from the real-time robotics simulation program Player/Stage. Performance analysis of FCE technique was done with existing four path planning algorithms under certain working parameters, namely computation time needed to find a solution, the distance travelled and the amount of turning required by the autonomous mobile robot. This study will enable readers to identify the suitability of path planning algorithm under the working parameters, which needed to be optimized. All the techniques were tested in the real-time robotic software Player/Stage. Further analysis was done using MATLAB mathematical computation software.

Keywords: Free-configuration Space, Eigenvector, Motion Planning, Trajectory Planning.
.

1. INTRODUCTION

Motion planning is one of the most important tasks in intelligent control of an autonomous mobile robot (AMR). It is often decomposed into path-planning and trajectory planning. Path planning is referred as to generate a collision free path in an environment with obstacles. Whereas, trajectory planning schedule the movement of a mobile robot along the planned path. Based on the availability of information about environment, the path-planning algorithms are divided into two categories, namely offline and online. Offline path planning of robots in environments uses complete information about stationary obstacles and trajectory of moving obstacles, which are known in advance. This method is also known as global path planning. When complete information about environment is not available in advance, the mobile robot gets information through sensors, as it moves through the environment. This is known as online or local path planning. Essentially, online path planning begins its initial path offline but switches to online mode when it discovers new changes in obstacle scenario. Classical approaches used in online path planning are Potential Filed approach (PF), collision–cone approach, and vector field histogram (VFH) method. Khatib [1] proposed the Artificial Potential Field (APF) approach which is popular in mobile robotics. This approach is known for its mathematical elegance and simplicity

as the path is found with very little computation. However, the drawback of this algorithm is that the robot may become stagnant or trapped when there is a cancellation of equal magnitudes of attractive and repulsive forces. Till date many variants of the potential field approach like escape-force algorithm [2], trap recovery model, adaptive virtual target algorithm etc. have been proposed. Path planning problems can also be solved by VFH approach [3]. At every instant, a polar histogram is generated to represent the polar density of obstacles around a robot. The robot's steering direction is chosen based on the least polar density and closeness to the goal. In a given environment, the polar histogram must be regularly regenerated for every instant and hence this method is suited for environments with sparse moving obstacles. Another commonly used online approach is based on the collision cone concept [4]. The Collision of a robot can be averted if the relative velocity of it with respect to a particular obstacle falls exterior to the collision cone. Another online approach for obstacle avoidance is dynamic windows approach [5]. The dynamic window contains the feasible linear and angular velocities taking into consideration the acceleration capabilities of a robot. Then the velocity at the next instant is optimized for obstacle avoidance subject to vehicle dynamics. With classical techniques, the optimum result requires more computational time due to incomplete information of the environment. This classical approach can be combined with heuristic approaches like genetic algorithm (GA) and particle swarm optimization (PSO) [6]. Another class of online path planning algorithms are sampling based path planning algorithms like rapidly evolving random trees (RRT) and probabilistic roadmap methods PRM [7]. The idea of connecting points sampled randomly from the state space is essential in both RRT and PRM approaches.

A limitation of the classical planning algorithm is that a complete model of the environment is needed before the planner can proceed. A solution to this problem is a sensor based path planner. Sensor based planner allows robots to work autonomously in unknown environments. Specifically the robot is able to move to a given configuration without prior knowledge of the environment. The motion of the robot is generated step by step while more and more knowledge about the environment is accumulated incrementally. A great variety of sensors that can be used for robots includes tactile sensor, vision sensor, sonar sensor etc. A well-known sensor based technique is "Bug" family. These path planners incorporates sensing and path planning as an integrated solution. Bug algorithms assume only local knowledge of the environment and a global goal. The most commonly used sensor based robot path planners are Bug1, Bug2, TangentBug, DistBug, and VisBug [8] .Bug 1 and Bug 2 uses tactile sensors while tangent bug, and Distbug uses range sensors. These techniques require its own position by using odometry, goal position and range sensor. In this paper, a novel sensor based path planning technique is proposed, namely, free-configuration eigenspace (**FCE)**. This approach tends to find principal components (Eigenvectors) spanning the low dimensional space (Eigenspace) of high order scanning data. Integrating the highest eigenvector in time steps can produce a collision free trajectory.

In this paper, we have compared our proposed FCE path planning approach with other well-known path planning techniques. Also, we have analyzed the path produced by any path planner based on the following parameters. **Path Length:** distance of the path from start to finish. **Computation time:** algorithm's total execution time excluding time spent during driving (i.e. from start to goal). **Turning:** the amount of turning which is performed along the path from start to finish. **Memory requirements:** the amount of global memory reserved by the algorithm. For good path planner it is assumed that all these parameter should be as small as possible.

This paper is organized as follows, in **section 2**, the Problem formulation and proposed technique is presented. In **Section 3** the materials and methods used are explained. **Section 3.3** describes the detection of eigensapce and trajectory generation algorithms. **Section 4** explains about the experimental set up and implementation of the FCE algorithm. In **section 5** a detailed performance analysis done with existing four planning algorithms namely, **APF, A*, RRT** and **PRM** and **section 6** with conclusion.

2. PROBLEM FORMULATION

2.1 Free-Configuration Space Concept

Finding a collision free path problem can be formulated as a search in a configuration space [9]: Let A be a robot, moving in a Euclidean space. $W = R^N$, where N = 2 or 3. Let $B_1, B_2, \ldots, B_n$ be fixed rigid bodies distributed in ***W*** are called as obstacles and are the closed subsets of ***W***. A configuration of ***A*** is a specification of the position of every point in ***A*** with respect to F_W, where F_W is a Cartesian coordinate system. The configuration space of ***A*** is the space denoted by C, with all possible configurations of ***A.*** The subset of ***W*** occupied by ***A*** at configuration *q* is denoted by ***A(q).*** A path from an initial configuration q_{init} to a goal configuration q_{goal} is a continuous map with $\tau : [0,1] \to C$ $\tau(1) = q_{goal}$ and $\tau(0) = q_{init}$. The workspace contains a finite number of obstacles denoted by $B_1, B_2, \ldots, B_n$. Each obstacle maps in ***C*** to a region:

$$C(B_i) = \{q \in C \mid A(q) \cap B_i \neq 0\} \tag{1}$$

Which is called as $C_{obstacle}$. The union of all $C_{obstacle}$ is the region $\bigcup_{i=1}^{n} C(B_i)$ and the set

$$C_{free} = C - \bigcup_{i=1}^{n} C(B_i) \tag{2}$$

A collision free path between two configurations is any continuous path $\tau : [0,1] \to C_{free}$

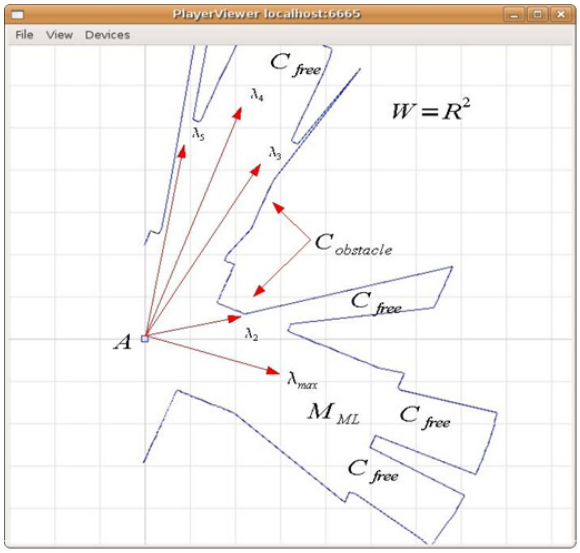


FIGURE 1: Configuration Space 2D Model - The blue line represents the laser scan area, whereas the red arrows inside the scan area are the eigenvectors and the red arrows outside the laser scan region show the obstacle region.

2.2 Proposed FCE Technique

To obtain a collision free path, we have utilized our paradigm [13] of "***Free-Configuration Eigenspaces***" which explains the underlying *low-dimensional manifolds of laser scanning data, represented by an eigenvector which corresponds to the free-configuration space and the vectorial combination of all these eigenvectors at discrete time scan frames manifests a trajectory.* To realize the above methodology of path planning, we have set forth the basis of two critical hypotheses:

1. H1: In the exploration process there lies a free-configuration space, such that the discrete paths which lie in that space can form an exploratory-trajectory

2. H2: The free-configuration space path shows a distinct pattern in the sensor data (surrounded by obstacles) and can be learnt to produce a better quality exploratory map.
According to the hypothesis:

- The single vector formulation of the trajectory point can be described as;

$$C_{free} \cap M_{ML} = C_{free}(E_{\lambda \max}) \tag{3}$$

- Consequently the vectorial sum of all the free-configuration eigenspaces;

$$\sum_{i=1}^{n} C_{free} \cap M_{ML} = C_{free}(E_{\lambda \max 1} + E_{\lambda \max 2} + E_{\lambda \max 3}) \tag{4}$$

3. MATERIALS AND METHODS

3.1 Laser Sensor Model

Range sensing is a crucial element of any obstacle avoidance system. Sensors suitable for obstacle detection are 2-D LADAR (i.e., a laser that scans in one plane). 2-D laser scanners are widely used sensor for obstacle detection. The SICK LMS is a laser scanner based on the measurement of time-of-flight (TOF) as shown in Fig.2 (a). The Laser Cartesian space is defined by its range and bearing, with the resolution of 0.5^0 , having 180^0 of frontal AMR scan with the range of 5 meters. Where the center position of the AMR is represented as (AMR_x, AMR_Y) in terms of the world coordinates frame. The incoming data are numbered as r_j , where $j = 0,1,\ldots,180$ are the distance value from the central position of the laser range finder to the object. The object position (Obj_x, Obj_y) is determined by measuring distance from the central position of the laser scanner to the object. The object position (Obj_x, Obj_y) is determined by the following where r_d is the distance from the centre of the SICK LMS to that of the mobile robot and θ_c is the angle of the mobile robot measured counter clockwise from the positive x-axis.

$$Obj_x = AMR_x + r_j \cos\left(\frac{j\Pi}{180} + \theta_c + \frac{\Pi}{2} - r_d \cos\theta_c\right) \tag{5}$$

$$Obj_y = AMRy - r_j \sin\left(\frac{j\Pi}{180} - \theta_c + \frac{\Pi}{2} - r_d \sin\theta_c\right) \tag{6}$$

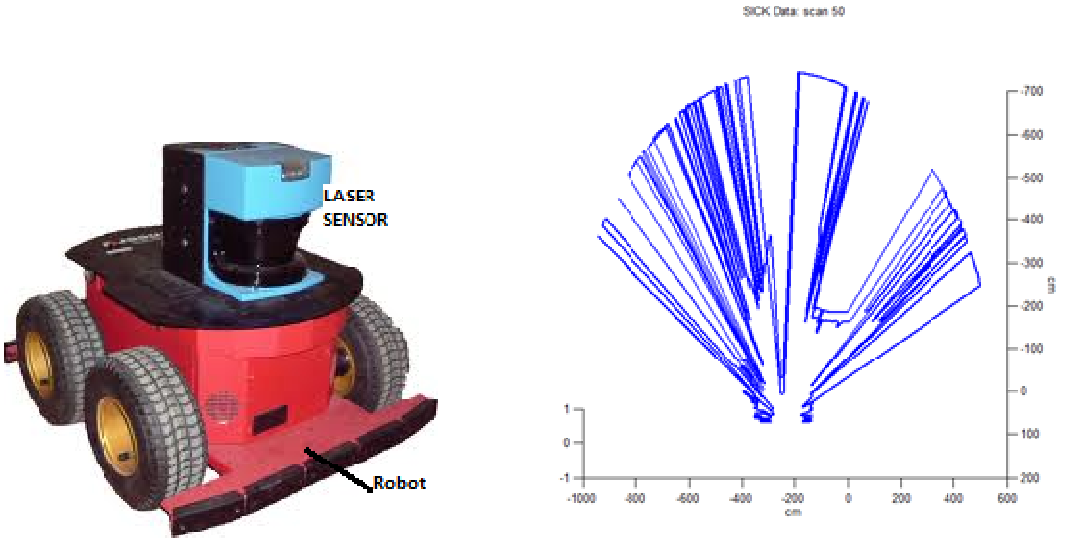


FIGURE 2: (a)The Pioneer 2AT autonomous mobile robot (AMR) carrying an onboard SICK laser scanner (b) A single laser scan plot output.

3.2 Dimensionality Reduction Technique

Using the SICK-laser a large number of point features can be obtained and all of them correspond to the same environmental structure. The dimensionality reduction technique enables to extract higher-order features in lower dimensional manifold representations which capture the data patterns. Principal Components Analysis (PCA) is a powerful tool that computes a set of bases functions that can be linearly combined to represent a collection of data [16]. PCA has been used extensively to cluster sets of point features in a map. We can extract line segments to represent each cluster of point features, generating very efficient representations, e.g., a representation using just the four parameters of the endpoints of 2-d line segment. In this case, the highest eigenvector is to be found by applying PCA to the sensor data which can be integrated in discrete time scan to get a trajectory.

3.3 Eigenspace Detection Method

It is possible to produce a trajectory as vectorial combination of highest eigenvectors in discrete time scan frame of sensor scanning data. But to ensure the easy manoeuvring of the robot along this direction shown by the eigenvector, one must make sure that there is enough space for the vehicle to pass through. Hence it can be ascertained by plotting the all the eigenvectors obtained by the dimensionality reduction technique, with reference to the vehicle pose at that point along with the sensor data. The pose of the mobile robot is denoted by the position and orientation (x, y, θ) in the map. In the beginning we compute the visibility model, which is the data2D laser sensor, which is commonly used for constructing range data maps in robotics. In order to construct such map, sensor's position and robot's orientation, while obtaining laser scans should be estimated accurately. The algorithm for eigenspace detection is described below. From the Fig. 3 it is evident that the highest eigenvector shows the direction of maximum free space through which the vehicle can easily manure without hitting the obstacle and hence this space we have named as free-configuration eigenspace.

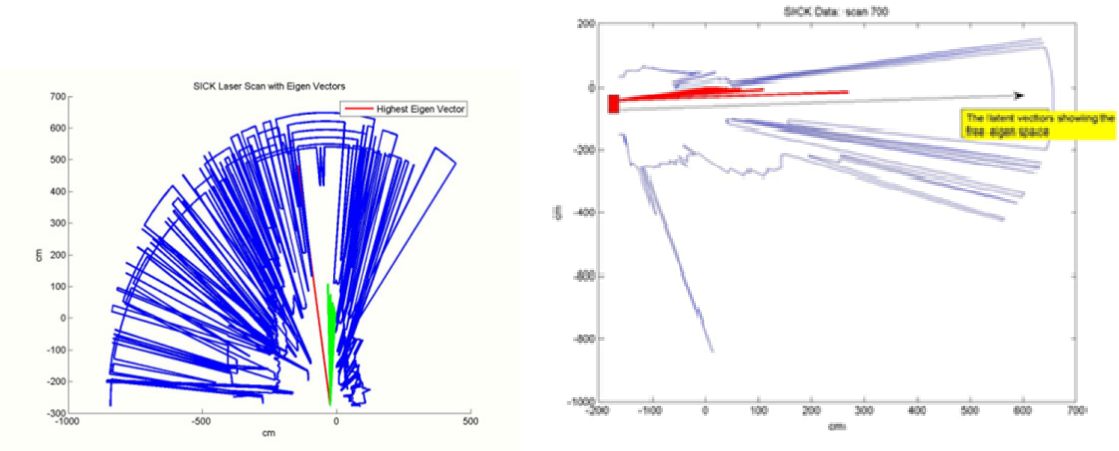
Algorithm 1: Finding Free-Configuration Space
Input: The Pose data (x, y, theta), sensor scanning data (matrix)
Output: Eigenspace (Eigenvectors)

Begin
Initialize Pose (x, y, theta), n, p
M = [ ]
for Time = 1 to n
for angle =-90 : 0.05: 90
M(time, angle) = d(Time, angle)
end
end
Calculate the Covariance of M

Calculate the Eigenvalues (V) and Vectors (E) of the Covariance of M
for i = 0,1,…,m
for angle =-90 : 0.5:90

$$E_{x\lambda} = E_{\lambda \max} \times \cos(angle)$$
$$E_{y\lambda} = E_{\lambda \max} \times \sin(angle)$$

end
end
Plot the eigenvectors from the corresponding vehicle pose,
End



(a) (b)

FIGURE 3: (a) Principle components extracted from the laser data, the green vectors are the smaller eigenvectors, whereas the red vector is the highest principal component, (b) the RED Vector indicating towards maximum the free area.

3.4 Autonomous Robot Trajectory Generation Method

According to the hypotheses in low-dimensional manifolds of laser scanning data, there lies an eigenvector which will represent free area or obstacle area. Assuming this to be free area, the highest eigenvector in discrete time can be integrated to get an exploratory trajectory .The

following algorithm describes trajectory formation using eigenvectors. The obtained eigenvector trajectory and the actual trajectory with laser scans for Navlab test bed data are shown in Fig.4.

Algorithm 2: Trajectory connectivity in Free-Configuration Space

Input: Robot Pose data, sensor measurement data M
Output: Highest Eigenvector Sensor data in discrete time scan
Begin
Initialise Pose (x, y, theta), n, p
M = [ ]
for Time = 1 to n
for angle =-90 : .05:90
M(time, angle) = d(Time, angle)
end
end
Calculate the Covariance of M
Calculate the Eigenvalues (V) and Vectors (E) of the Covariance of M
Sort the diagonal of E in descending order
Taking the Maximum of E and its corresponding V
Plot E and V from the initial Pose(x,y,theta) as follows

$$E_{x\lambda} = E_{\lambda \max} \times \cos(\ angle\ )$$
$$E_{y\lambda} = E_{\lambda \max} \times \sin(\ angle\ )$$

Plot the eigenvectors from the corresponding vehicle pose ,Pose(x,y,theta)
End

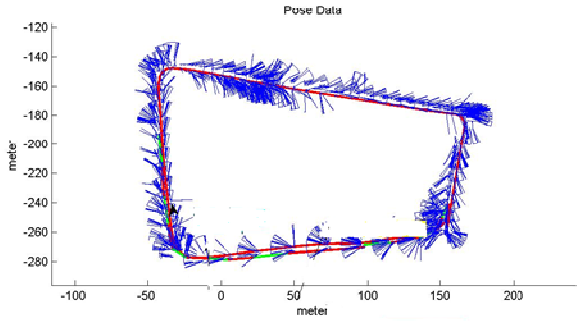


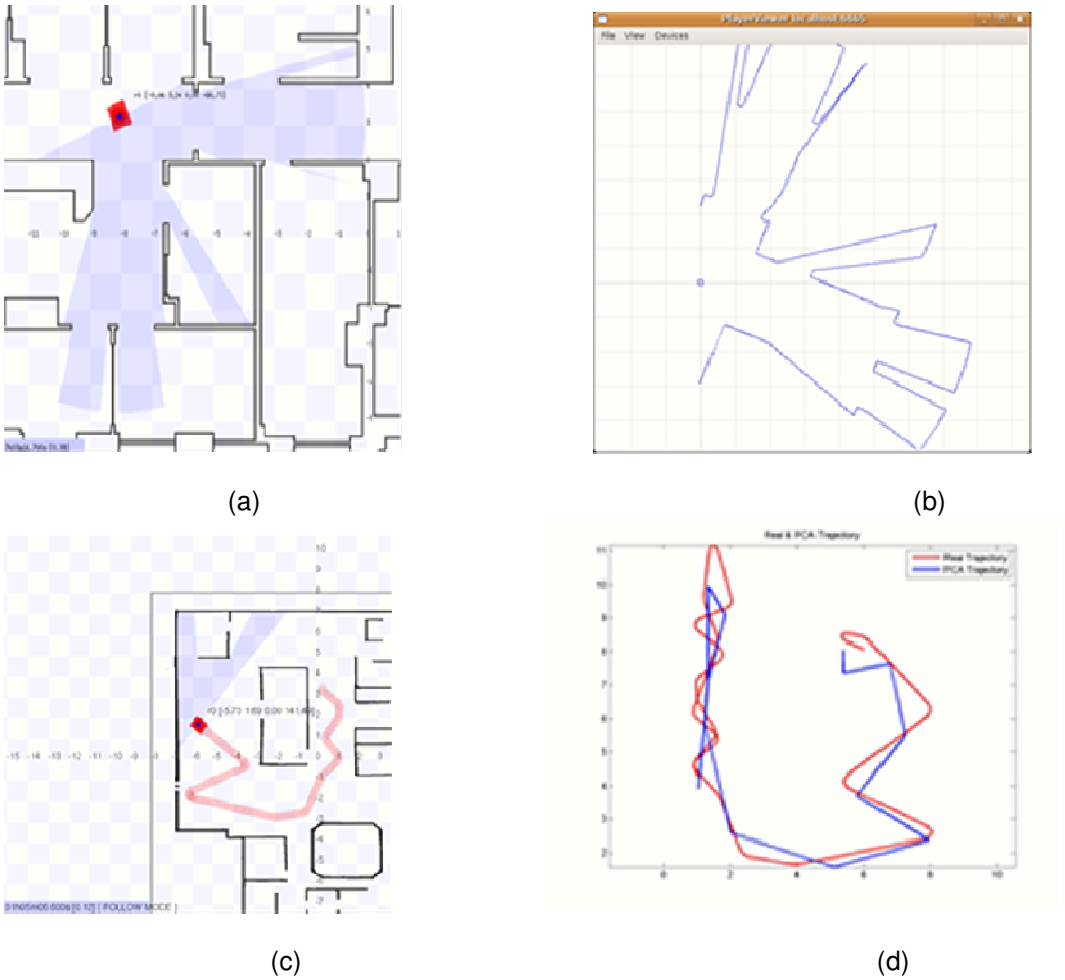
FIGURE 4: Green & red lines showing the trajectory formed by the Eigenvectors in discrete time scan of laser data.

4. EXPERIMENTAL SETUP and SYSTEM DESCRIPTION

4.1 Player/Stage Data Simulation

The proposed technique was applied to the robotic simulator running under real-time operating system (RTOS) Ubuntu 8.04 Hardy–heron on 2.0 GHz Intel dual core processor having 3 GB of RAM. The environment was created in real time (distributed under GNU) 2D autonomous mobile robotic software, Player/Stage as shown in Fig. 5. A configuration definition XML file was created that contain positions of environment floor plan, a pioneer 2DX autonomous mobile robot, laser sensor, odometry sensor. The mobile robot is defined as a non-holonomous (pioneer2DX) , which can move around in the environment and can sense the obstacle by measuring the laser distance

from its Centre point of gravity. TCP/IP sockets are used to for the communication between the mobile robotic agent and the robotic software server. For experimental purposes, the mobile robot's Laser data and the odometer readings was recorded using player/stage data commands, processed/ reduced online as shown in the Fig. 5.The proposed trajectory detection methodology was implemented in MATLAB (product of Mathworks Inc.). The actual trajectory was divided into segments and each segment's consecutive laser sensor readings were taken. The latent values were found for these chunks of laser data using PCA analysis. The latent values (highest eigenvector) were integrated in time and the new trajectory was obtained. The scenario of the robot and environment is shown in the Figure.5(c) and the actual trajectory (in red) and the eigenvector trajectory in blue



(a) (b)

(c) (d)

FIGURE 5: (a) 2Dview of the environment with the laser mounted autonomous mobile robot. (b) Simulated laser plot. (c) Graphical User Interface client program "playerv" that visualizes laser data from a player server, (c) The Actual trajectory & learned trajectory (red).

4.2 Real Time Data Simulation

The scenario of Navlab SLAMMOT Datasets, Carnegie Mellon University USA as shown in Fig. 6 in yellow Loop-2. The laser data for the entire loop was divided in to four segments and within each segment consecutive laser sensor readings was taken. The latent values were found using PCA analysis and the latent values (highest eigenvector)were integrated in time and the new trajectory was obtained .This new trajectory and the actual trajectory was compared and an error graph was plotted as shown in Fig. 6 (c , d, e, g) . It was concluded from the Fig. 6(b) that at the

corners, the direction of eigenvector obtained deviated from the actual trajectory where as in a straight line segment, the eigenvector is very much close to the actual trajectory.

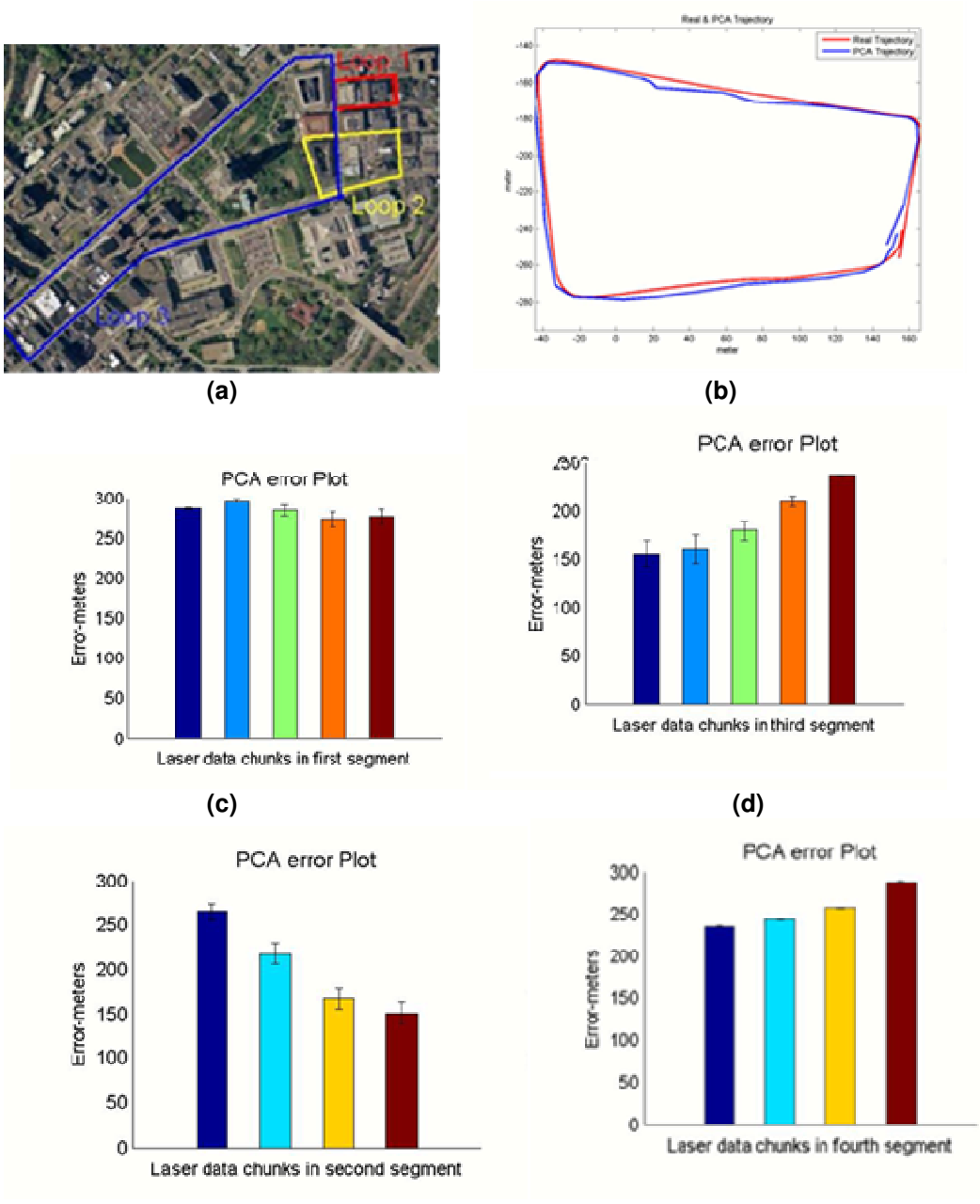


FIGURE 6: (a)The Aerial Photo is downloaded from and the copyrighted property of GlobeXplorer, LLC. ("GlobeXplorer"). The trajectory (Loop-2) in yellow, was used for laser data-set collection. (b)The trajectory obtained after applying PCA to the laser data-set. (c).(d),(e) (f)error-plots show the respective trajectory segments vs laser-data chunks and their error bounds for PCA with real-trajectory.

4.3 Experimental Result Analysis

We have tested the developed algorithm on real time and simulated laser data set. From the result plots for the real time data, as shown in Fig. 6(b) & Fig. 6(d), it is evident that the constructed eigenvector trajectory (in red) is perfectly matching on the longer segments of the actual trajectory. But, failed to produce good results on the corners. In order to get a clear picture

about the corner misalignment of the trajectory, we plotted the Navlab data trajectories with in the 2D map of the laser data. It is also evident from Fig. 7, at the corners the eigenvector trajectory is deviating away from the actual trajectory of the robot.

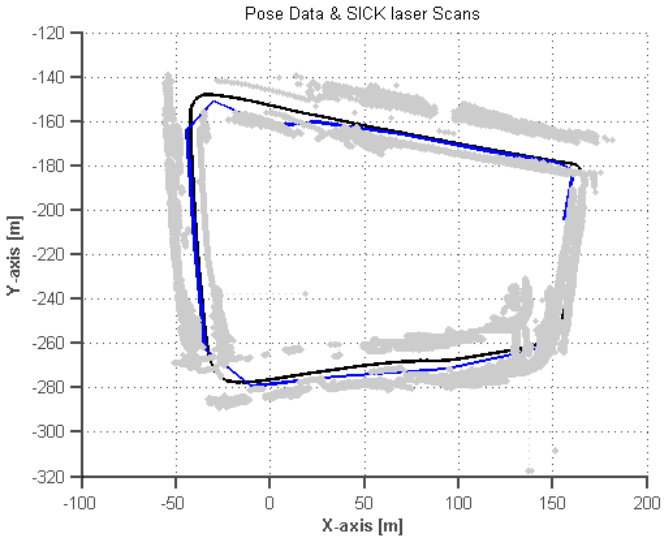
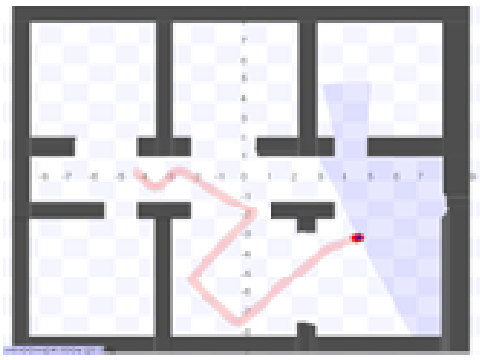


FIGURE 7: The 2D map obtained from plotting the outliers of laser data and the actual trajectory (black) and the eigenvector trajectory(blue).

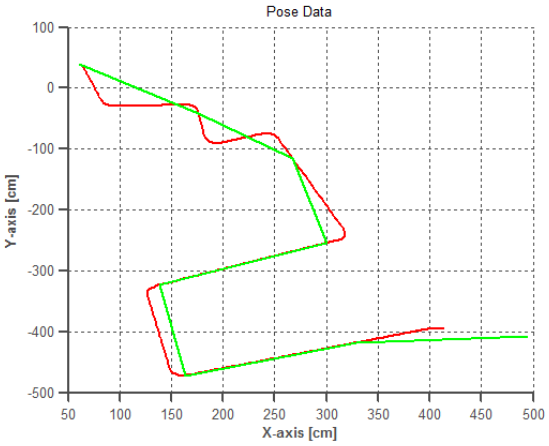
5. PERFOMANCE ANALYSIS

The path produced by any path planner can be analysed on the following parameters. **Path Length**: distance of the path from start to finish. **Computation time:** algorithm's total execution time excluding time spent driving. **Turning**: the amount of turning which is performed along the path from start to finish. **Memory requirements**: the amount of global memory reserved by the algorithm. The following section illustrate the result performance analysis done existing path planners **APF**, **RRT**, **PRM** and **A*** with the proposed **FCE** method. We have considered two scenarios Room1 and Room2 and found the estimated values for all the parameters under consideration. The analysis was done using MATLAB mathematical computation software. The below tables summaries the results obtained:

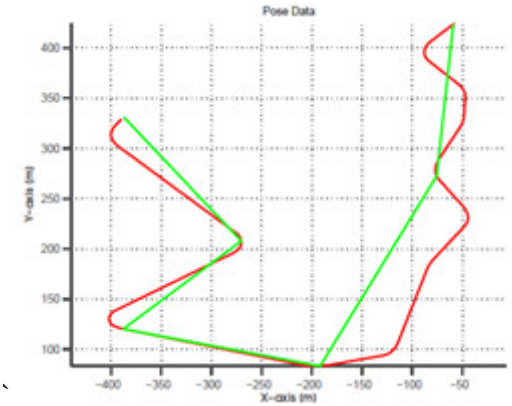


(a)

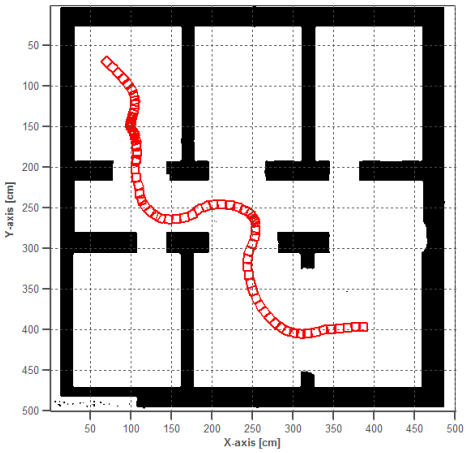
(b)


Pose Data
Y-axis [cm]
X-axis [cm]

(c)


Pose Data
Y-axis [m]
X-axis [m]

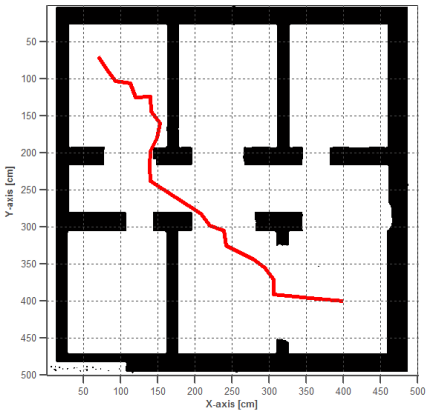
(d)


Y-axis [cm]
X-axis [cm]

(e)


Y-axis [cm]
X-axis [cm]

(f)


Y-axis [cm]
X-axis [cm]

(g)


Y-axis [cm]
X-axis [cm]

(h)

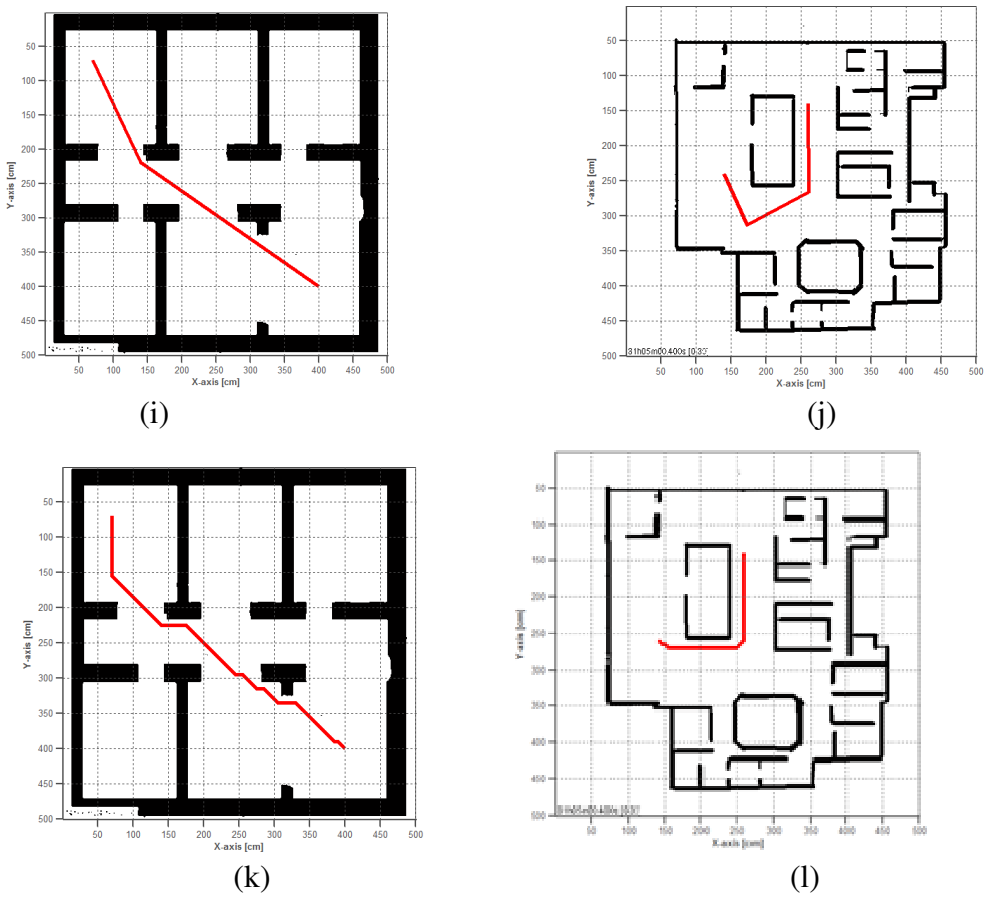


FIGURE 8: Trajectories obtained from different Path-planning alogrithms. a) Scenario of Room-1. b) Scenario of Room-2. c) Trajectories obtained from FCEalgorithm from scenario of Room-1. The red is normal way points trajectory, where as green is the FCE algorithm trajecorty. d) Trajectories obtained from FCE algorithm from scenario of Room-2. The red is normal way points trajectory, where as green is the FCE algorithm trajecorty. e) Trajectory obtainedfrom APF method for scenario of Room-1. f) Trajectory obtained from APF method forscenario of Room-2. g) Trajectory obtained from RRT method fromscenario of Room-1. h) Trajectory obtained from RRT method from scenario of Room-2. i) Smoothed trajectory obtained from PRM method from scenarioof Room-1. j) Smoothed trajectory obtained from PRM method from scenario of Room-2. k) Trajectory obtained from A* method from scenario of Room-1.l) Trajectory obtained from A* method from scenario of Room-2.

Source = [70 70]; in Y, X format.
Goal = [400 400] in Y, X format.

Technique	Scenario	Path length(cm)	Processing time	Max-turn(cm^{-1})
RRT	Room1	5.646315e+002	1.513718e+002	0.1412
PRM	Room1	4.491235e+002	1.848314e+001	-0.0019
APF	Room1	5.889301e+002	1.846913e+000	3.00
A*	Room1	4.740559e+002	4.430567e+001	0.1474
FCE	Room1	10.91235e+002	1.98314e+001	1.7471

TABLE 1: Result for Room1.

Source = [140 260]; in Y, X format.
Goal = [260 140] in Y, X format.

Technique	Scenario	Path length(cm)	Processing time	Max-turn(cm^{-1})
RRT	Room2	3.347247e+000	3.012290e+000	0.0004
PRM	Room2	2.806335e+002	3.273510e+000	0.0854
APF	Room2	3.234583e+002	1.037137e+000	-0.0204
A*	Room2	2.482843e+002	5.420330e+001	0.1574
FCE	Room2	6.234583e+002	2.012290e+000	2.900

TABLE 2: Result for Room2.

5.1 Result Analysis

As shown in Table-1 & 2, performance analysis on different path planning algorithms shows that the PRM technique performs better in terms of turning and path length. But it is probabilistic complete. But RRT is faster as compared to PRM and produce fine path with minimum turning. Even though A*shows an optimal path, the computational cost is high and the clearance space from the obstacle is low. The APF algorithm requires less computational time but the path length depends on the set value of the potential forces and it suffers from local-minima problem. In case of FCE, the path length and turning value are comparatively larger than all other methods.A good path is relatively short, keeps some clearance distance from the obstacles, and is smooth. Result analysis shows APF and proposed FCE technique is better on this attributes.

6. CONCLUSIONS

This paper proposes a novel path planning problem of an autonomous mobile robot navigating in a structured unknown environments, using free-configuration eigenspaces (FCE). The results are very much consistent with the hypothesis we laid for our research in the beginning, i.e. trajectory formation in correspondence to the highest eigenvector of the free-configuration laser data always results into a better exploration of the unknown area. Consequently by adapting to the similar trajectory formations, the autonomous mobile robot has better tendency towards the map-building process. The trajectories maximizing the map building process could be formed by the intramural property of the laser sensor dually responsible for the map building process, the vector sum of all these highest vectors obtained from the laser data result into the trajectory which maximizes the information of the map. These individual eigenvectors were machine learned and the predicted new trajectory vectors facilitated the autonomous robot to maneuver at considerably higher speed. A limitation of our proposed approach is the assumption of simple obstacle geometry. This can be tackled by situation aware sensing descriptors, which may result into dimensionality scaling and require algorithms that scale well.

7. REFERENCES

[1] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” The international journal of robotics research, vol. 5, no. 1,pp. 90–98, 1986.

[2] P. Vadakkepat, T. H. Lee, and L. Xin, “Application of evolutionary artificial potential field in robot soccer system,” in IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th, pp. 2781–2785, IEEE, 2001.

[3] J. Borenstein and Y. Koren, “The vector field histogram-fast obstacle avoidance for mobile robots,” IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278–288, 1991.

[4] A. Chakravarthy and D. Ghose, “Obstacle avoidance in a dynamic environment:A collision cone approach,” IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 28, no. 5, pp. 562–574, 1998.

[5] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23–33, 1997.

[6] P. Vadakkepat, K. C. Tan, and W. Ming-Liang, “Evolutionary artificial potential fields and their application in real time robot path planning,”in Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, vol. 1, pp. 256–263, IEEE, 2000.

[7] D. Gallardo, O. Colomina, F. Fl´orez, and R. Rizo, “A genetic algorithm for robust motion planning,” in Tasks and Methods in Applied Artificial Intelligence, pp. 115–121, Springer, 1998.

[8] S. Tang, W. Khaksar, N. Ismail, and M. Ariffin, “A review on robot motion planning approaches,” Pertanika Journal of Science and Technology, vol. 20, no. 1, pp. 15–29, 2012.

[9] H. M. Choset, Principles of robot motion: theory, algorithms, and implementation. MIT press, 2005.

[10] E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd, and E. Kavraki, “Sampling-based roadmap of trees for parallel motion planning,” IEEE Transactions on Robotics, vol. 21, no. 4, pp. 597–608, 2005.

[11] R. Al-Hmouz, T. Gulrez, and A. Al-Jumaily, “Probabilistic road maps with obstacle avoidance in cluttered dynamic environment,” in Intelligent Sensors, Sensor Networks and Information Processing Conference, pp. 241–245, IEEE, 2004.

[12] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” IEEE Transactions on Robotics and Automation, vol. 12, no. 4, pp. 566–580, 1996.

[13] T. Gulrez, S. Zaheer, and Y. Abdallah, “Autonomous trajectory learning using free configuration-eigenspaces,” in IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), pp. 424– 429, IEEE, 2009.

[14] T. Gulrez and A. Tognetti, “A sensorized garment controlled virtual robotic wheelchair,” Journal of Intelligent & Robotic Systems, vol. 74, no. 3-4, pp. 847–868, 2014.

[15] T. Gulrez, A. Tognetti, and D. De Rossi, “Sensorized garment augmented 3d pervasive virtual reality system,” in Pervasive Computing, pp. 97– 115, Springer, 2010.

[16] I. Jolliffe, Principal component analysis. Wiley Online Library, 2005.

[17] A. Al-Odienat and T. Gulrez, “Inverse covariance principal component analysis for power system stability studies,” Turkish Journal of Electrical Engineering & Computer Sciences, vol. 22, no. 1, pp. 57–65, 2014.

[18] M. Tipping and C. Bishop, “Probabilistic principal component analysis, “Journal of the Royal Statistical Society”, Series B, vol. 61, pp. 611–622, 1999.

[19] “Player stage.” http://playerstage.sourceforge.net/.

[20] T. Chaudhry, T. Gulrez, A. Zia, and S. Zaheer, “Bezier curve based dynamic obstacle avoidance and trajectory learning for autonomous mobile robots,” in Proceedings of the 10th International Conference on Intelligent Systems Design and Applications, ISDA’10, 2010, pp. 1059–1065.

[21] S. Zaheer, T. Gulrez, “Beta-eigenspaces for autonomous mobile robotic trajectory outlier detection,” in 2011 IEEE Conference on Technologies for Practical Robot Applications, TePRA, pp. 31–34, 2011.

[22] P. Raja and S. Pugazhenthi, Review Optimal path planning of mobile robots: International Journal of Physical Sciences Vol. 7(9), 23, pp. 1314 – 1320, February, 2012

[23] R. Kala, K. Warwick Multi-Vehicle Planning using RRT-Connect. Paladyn Journal of Behavioural Robotics, 2(3): 134-144, 2011