Chapter 5

Approach for Path Planning and Tracking of Shape Aware Mobile Robot in Structured Environment using Vision Sensor

5.1 Introduction

Vision based obstacle detection and avoidance to reach to the final goal point is becoming a popular alternative to sonar- and infrared sensors, as it provides a better resolution at reasonable price (Ahmed et al., 2018; Mukhtar and Xia, 2015). Various concurrent technologies, which consider various aspects including environmental uncertainties, modeling errors and optimal path planning have been challenged through mobile robotics research. Considering the current availability of many sensors and high-performance processors, vision-based systems provide rich information about the real-time environment. Cameras used for vision-based obstacle detection provide more comprehensive information about the environment. Most computer vision methods that estimate range information based on camera images require a well textured and real-world environment to perform the task properly. Path planning usually refers to the problem of taking the solution from a robot motion planning and determining how to move the mobile robot along the way that favors mechanical limitations of the robot. Based on the priori knowledge of the environment, path planning of a mobile robot is to carry out a collision-free path from starting position to the target position, optimizing a performance criterion such as distance, time.

The path planner's prime objective is to navigate a collision-less path among obstacles in the environment [Kamkarian and Hexmoor (2015)]. However, real-time path planning problems are often too large to solve optimally within an acquired time. Moreover, even if an optimal path is found initially, the model used to represent the problem is unlikely to be perfect and susceptible to change in the environment because of obstacle and therefore a shape-aware factor may be introduced to incorporate the discrepancies in its model while executing plan. Shape aware factor focuses more on semantic relations among the parts of shape which computes geometrical features such as edge curvature rather than on their virtual geometry. The A∗ algorithm mostly generates a path that a mobile robot cannot follow because it does not consider the shape and size of the mobile robot. For real time implementation, the dimension of mobile robot is necessary. Thus, geometrical information related to the size of the obstacle and the size of robot will help the robot become aware of the environment while performing a task. The size awareness is considered as per the geometry of mobile robot and Obstacles present in the environment. This nomenclature has been taken from literature that discusses the perception of size-aware techniques (Moon at al. 2013, Withey et al. 2020).

In such scenario, the above factor needs to update its model and re-plan. In the grid environment, the nodes obtained by A* algorithm are connected in sequence to get the planned path for mobile robot, sometimes two or more obstacles appear consecutively on the route through which mobile robot cannot pass though. But A* algorithm shows the optimal path through that route and it is apparent that the planned path is not the desired path. The key steps involved in the functioning of the A* algorithm with respect to path planning have been explained in chapter 2. Thus, current chapter proposes a modified optimal path after which curve smoothing method is applied to execute the motion for completion of task.

Obstacle detection in real time scene is one of the most challenging problems, as it is necessary to deal with difficulties such as viewpoint changes, occlusions, illumination variations, background clutter or sensor noise and above all computational complexity (Lin et al., 2010). Substantial amount of work regarding vision sensor in feedback control has been carried out for applications such as: autonomous driving, manipulation, mobile robot navigation and surveillance. There has been research concern to generate collision free paths consists of obstacle in static or dynamic environment which is important for autonomous mobile robots to move towards a goal (Rios and Chaimowicz, 2010). The problem of path planning for a mobile robot moving in a two-dimensional scene occupied with obstacles with continuous boundaries and finite size is a real challenge. The information available corresponding to own current coordinates and those of the target position for the mobile robot is limited. The research work discusses the solution methodologies to avail the use of vision sensor optimally to perform a task by selecting path if the geometric structure of the scenes known. A geometric method has been used to obtain the dimensions of the obstacles in the environment without adding the outline of the mobile robot. Consequently, a shape aware A* algorithm has been proposed whose performance has been demonstrated via a series of simulations that re-plan a collision free path by considering the shape of mobile robot. The proposed method is tested in a real time environment and has been proven that the algorithm can move a mobile robot in a path that minimizes the deviations in planned path to reach the goal.

Robot Operating System (ROS) is a software framework that has been gaining traction at an astonishing rate since its release which handles information passing between various functions in the form of topics that publish information to which any other function can subscribe [Riedel and Franchi, (2012)]. In this work, ROS and MATLAB Bridge is presented that allows the integration of ROS into the MATLAB environment to conduct experiments with real-time mobile robot platforms [Galli et al. 2017, Das et al. (2020)]. Obstacle detection system incorporates vision-based refinement functions i.e. thresholding, erosion and clustering, with standard illumination. For this purpose, OpenCV open source vision library has been utilized which is implemented in different C++ and Python, mainly aimed at real time computer vision. OpenCV has been integrated into ROS which provides real-time image conversions between OpenCV and ROS Bridge formats. In this work, the 2D mobile robot detection and tracking is implemented using the computer vision library available in OpenCV. The library available in OpenCV provides implementations for different computer vision algorithms [Bradski and Kaehler (2008)], creating an API that could yield powerful results with simple code. It has the advantage of real-time detection along with a precise definition of the obstacles position and shape. The vision-based refinement is performed on demand through ROS in real-time. Therefore, obstacles are accurately detected, independent of their texture and shape. Furthermore, the precision of experimental approach is valuable for selecting the correct obstacle avoidance strategy. The idea is to use a ROS for automatic exploration and mapping on structured real-world environment.

The automatic obstacle detection, which consists of a gray conversion over the RGB color space, is performed only once before initiating the tracking. Therefore, the object should be in the field of view of the vision system at the beginning of the tracking process. The detected obstacle is subsequently used by the rectangular object detection methods which along with the threshold, erode and clustering functions run continuously to track the object. Since the obstacle detection approach itself is computationally fast, without the need of manual intervention in real-time environments where the light intensity changes frequently, causing the apparent color of the object to change. In order to experimentally validate the proposed method, the KLT tracking algorithm was implemented with Kalman filtering algorithm that address the real-time robotic visual tracking in 2D space where height of the obstacles is unknown. As KLT algorithm has limited capability like sparse optical flow process, thus Kalman filter is used to improve the accuracy and refine the visual tracking estimation results.

The rest of this chapter is structured as follows. In Section 5.2 a brief overview procedure for path planning (A*) Algorithm with respect to shape aware factor is explained. The system overview concerning a two-computer network for real time implementation is discussed in Section 5.3, Kinematic model of Mobile Robot with mathematical details are discussed in section 5.4. In Section 5.5, the real-time implementation is tested by creating different obstacle scenarios and results of these experiments have been presented. In section 5.6 with this a simple implementation of visual tracking-based control of mobile robot while following different trajectories are presented. Finally, the work is concluded in section 5.7, with the major findings and necessary direction for future work.

5.2 System Description

The experiments have been conducted using a three wheeled mobile robot. The mobile robot consists of two independently driven rear wheels and one omnidirectional wheel in the front without power. The mobile robot has a size of 30×30 cm^2 and the distance between the rear wheels is 175 mm. The mobile robot is shown in Figure 5.1. The rear wheels are mounted on servo motors which has diameter of 106 mm as diameter. The environment (the scene) in which the mobile robot moves is defined in a two-dimensional plane. A vision sensor (iball roboK20) is mounted overhead at a distance of 365cm from ground to track the robot. The images are acquired with the help of this sensor with a resolution of 640×480 pixels. The overhead camera is connected to a laptop which is kept away from the mobile robot for computational framework using Robot Operating System (ROS). This laptop has 1.7 GHz Intel Core i3-4005U processor and Intel HD Graphics 4400 and has Window 10, operating system. The rear wheels are driven by Dynamixel EX-106. These digital servomotors have no load speed equal to 69.9 rpm at 18.5V. These motors are actuated by onboard tablet computer Surface-3 which has 1.6-GHz quad-core Intel Atom processor, with 4GB RAM and 64GB onboard storage. In this tablet MATLAB software is installed to send commands for actuation of the motors. Dynamixel Ex-106 with a unique ID is controlled by serial packet communication on a BUS and supports network such as RS485 asynchronous serial communication. These EX-106 digital servomotors use USB2Dynmixel converter with PC to control. The mobile robot has two independent wheels that can be controlled independently and capable of rotating about the center of gravity (COG) of the robot chassis.

To facilitate smooth implementation ROS middleware environment is used. In this work the detected obstacles data has been sent to ROS environment for path planning algorithm. The scene observed is set up in 2D planes patterns, from which obstacles are extracted and matched to estimate the current and target images. The information about the obstacles comes from a simple vision sensor whose capability is limited to detect an obstacle. The main challenge is to implement path-planning algorithm with shape awareness as an integral part. In this sense, shape awareness is an extremely relevant property for robot navigation, given the impact of environmental changes on the performance of mobile robot. In this work it has been demonstrated the usage of ROS to improve the configuration planning capabilities of navigation task in a real world testing environment. Often in robotics, shape awareness refers to the ability of robots to sense and react according to the obstructions in the environment.

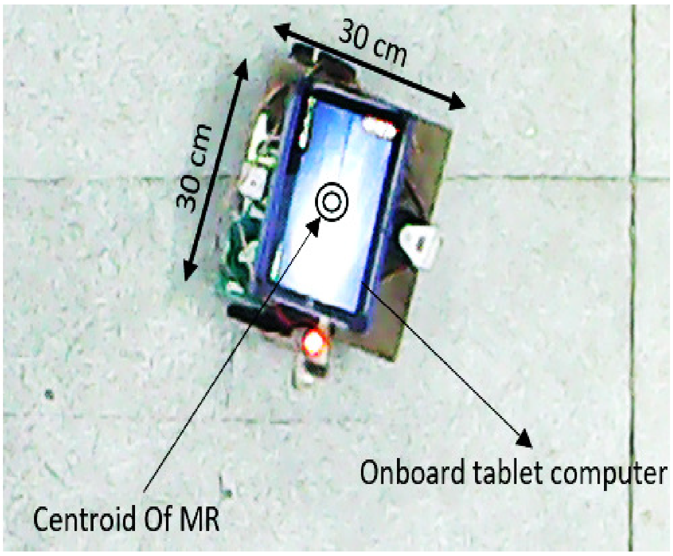


Figure 5.1: The mobile robot for the experiments

Particularly knowledge of mobile robot shape is important along with the shapes of obstacles while planning for path.

5.3 Kinematic Model of Mobile Robot

In this section, kinematic model for the designed mobile robot has been discussed. The mobile robot has desired capability to avoid obstacles in term of narrow space when wheels of the robot are in contact with ground all the time. The mobile robot shown in Figure 5.2 uses a differential drive steering. The kinematic model of the two wheeled mobile robot accommodates two input, linear velocity (V_L, V_R) and angular velocity $(\omega_L\ ,\ \omega_R)$ that is $(V_L = r\omega_L, V_R = r\omega_R)$ while moving toward the target position. The rate of change of position of robot in x-direction is $\dot{x}$ and that in y-direction is $\dot{y}$ and the angular velocity of the robot is $\dot{\theta}$ respectively in discrete time described in Equations 5.1 to 5.3:

$$\dot{x} = v\cos\theta \tag{5.1}$$

$$\dot{y} = v\sin\theta \tag{5.2}$$

$$\dot{\theta} = \omega \tag{5.3}$$

The v and ω are obtained from the variations in the speed of the right wheel (v_R) and the left wheel (v_L) simultaneously as shown in Equation 5.4 to Equation 5.6:

$$\dot{x} = \frac{(v_R + v_L)}{2}\cos\theta \tag{5.4}$$

$$\dot{y} = \frac{(v_R + v_L)}{2} \sin\theta \tag{5.5}$$

$$\dot{\theta} = \frac{(v_R - v_L)}{2} \tag{5.6}$$

where, r denotes the robot's wheel radius, and L denotes the distance between right wheel and left wheel (axle length). Thus, using Equations 5.1 to 5.6, v and ω which give the relation between mobile robot velocity to wheel velocity are calculated and shown in Equations 5.7 and 5.8.

$$v = \frac{r}{2}(\omega_R + \omega_L) \tag{5.7}$$

$$\omega = \frac{r}{L}(\omega_R - \omega_L) \tag{5.8}$$

Because the rate of rotation $\omega(t)$ about the ICC (Instantaneous Center of Curvature) must be the same for both wheels, it is described in Equations 5.7 and 5.8.

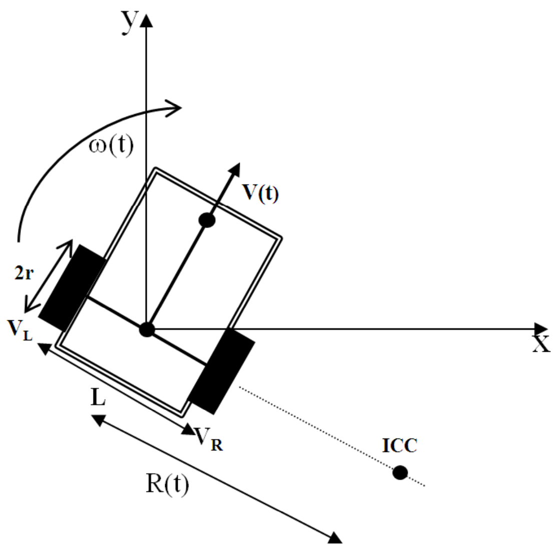


Figure 5.2: Kinematic model

Here R denotes radius of the curve (measured along the midpoint between the wheels). At any instant in time geometry can be used to solve for R and the equation becomes

$$R = \frac{L}{2}\left(\frac{V_R + V_L}{V_R - V_L}\right) = \frac{L}{2}\left(\frac{\omega_R + \omega_L}{\omega_R - \omega_L}\right) \tag{5.9}$$

In matrix form mobile robot velocities are represented as:

By varying the velocities of the two wheels, the robot can be taken with varying trajectories. The arcs length ΔD traveled by the two driven wheels and mobile robot rotates by $\Delta\theta$ degrees are calculated by

$$\Delta D = \frac{\Delta r_L + \Delta r_R}{2} \tag{5.11}$$

$$\Delta\theta = \frac{\Delta r_L - \Delta r_R}{2} \tag{5.12}$$

Thus, change in position and orientation in time Δt is shown by

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta D \cos \Delta\theta \\ \Delta D \sin \Delta\theta \\ \Delta\theta \end{bmatrix} \tag{5.13}$$

5.4 Overview of Experimental Implementation

Accurate path planning is very important for the mobile robot to be able to track an optimal path between start position and goal position avoiding collision with obstacles. In this section, the steps followed are summarized regarding the detection of obstacle and path planning with shape awareness of mobile robot.

Step 1: Path planning- Based on the current knowledge of the environment (mapping, detection of obstacles), a path between the current node to the target node will be planned using A* path planning algorithm. In this case the obstacles are treated based on cell which cannot be accessed (not to be used without actual dimension of the obstacles).

Step 2: Sensing- The position of obstacles will be detected with the help of sensor data. Here the actual dimensions of obstacles are found out by using Thresholding, Erosion, Clustering functions. If there is no feasible path found because of the obstacles then go to step 3, else go to step 4.

Step 3: Re-planning of the path- The path is planned again using A* algorithm using the dimensions of the obstacles. Since the path has not considered the dimensions, the A* path planning algorithm will be used again to modify and obtain the revised path. If there is no feasible path found in presence of the obstacles with desired dimensions, then STOP the path planning algorithm and display there is no path.

Step 4: Fine tuning of the path- The preplanned path obtained in step 3, may not be executed by the motors because of holonomic constraints and thus the sharp corners or bends of the path will be splined to execute smooth motion.

Step 5: Motion control-With the revised path, the digital motors of mobile robot will be feed with required inputs to follow the desired path.

Step 6: Detection and tracking- During the execution of task, the features of mobile robot will be detected by vision sensors through Viola-Jones algorithm. For the purpose of tracking the mobile robot, KLT algorithm will be used.

Step 7: Kalman filtering- To remove the noise in tracking of mobile robot, Kalman Filter will be used.

Present work integrates obstacle detection mechanism with the path planning algorithm to make the mobile robot shape aware and use a modified path as compared to path obtained from A* algorithm. This algorithm to search for the obstacles in real-time scenarios and modifies the path with vision sensor related information. The flowchart for implementation of shape aware integrative method has been described in Figure 5.3.

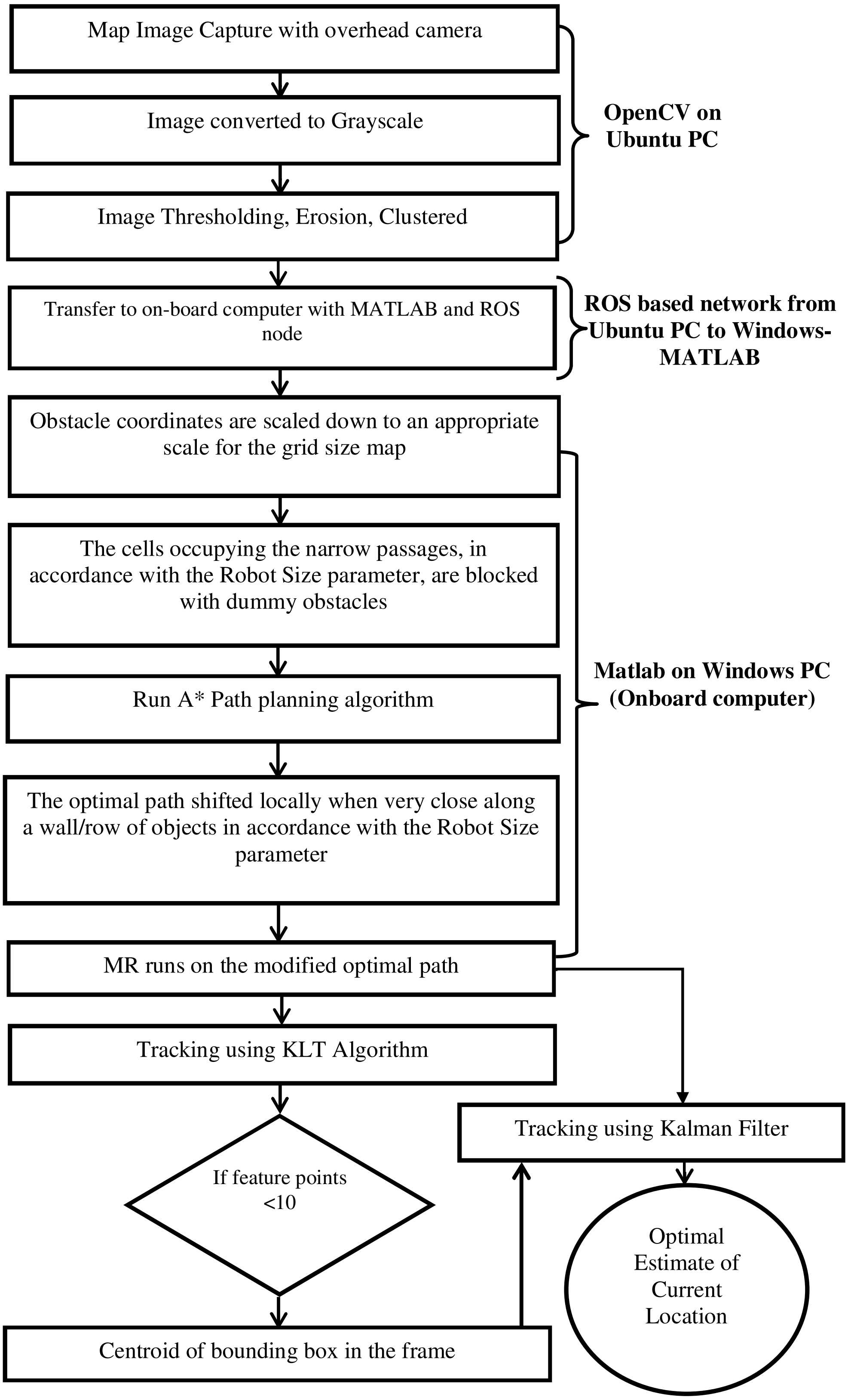
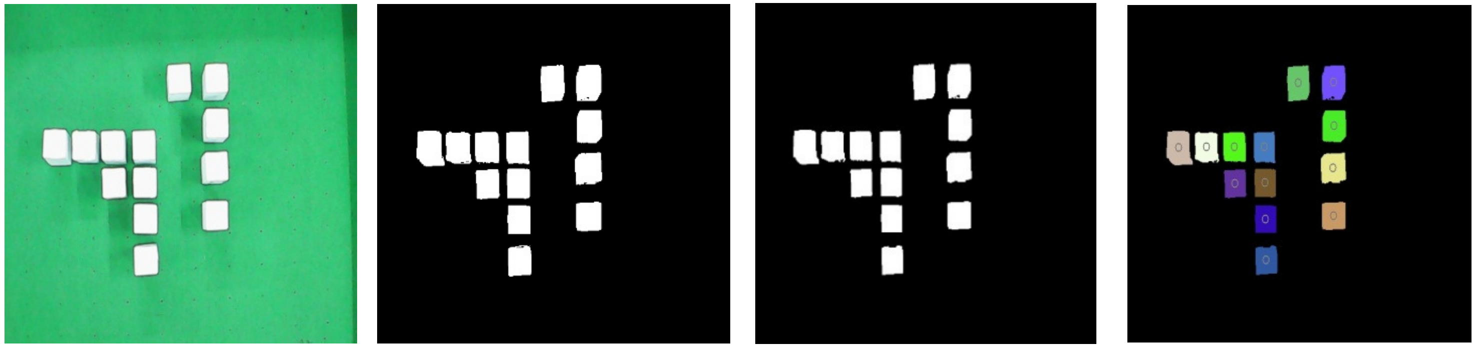


Figure 5.3: Flow chart for path planning and tracking of shape aware mobile robot

5.5 Experimental Results and Analysis

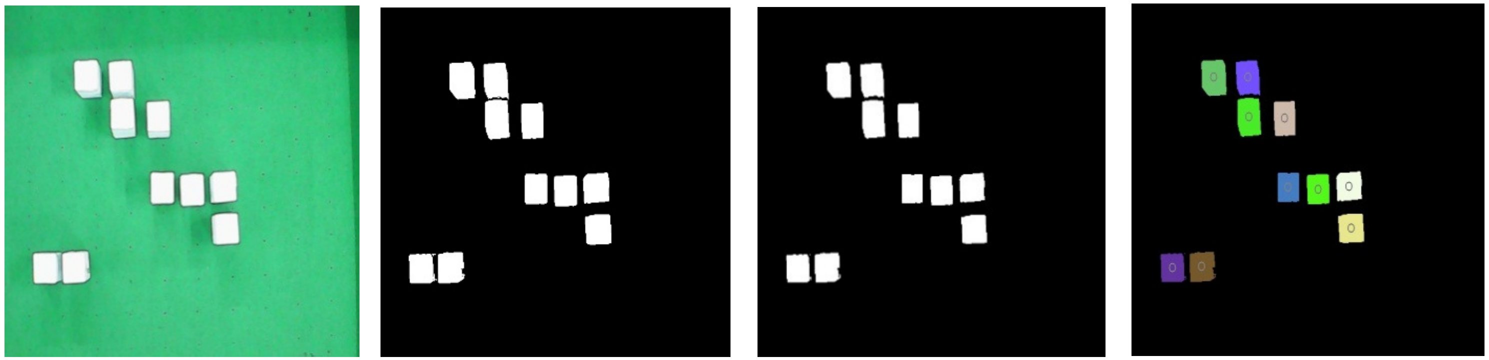
The investigation has been carried out for the mobile robot that explores a real-time environment by relying on its vision sensor. The steps used for the purpose involve calibration of vision sensor, path planning and construction, detection and tracking. The methodology has been implemented on the mobile robot in real-time, which has an overhead vision sensor (camera) for map generation, obstacle detection and real-time mobile robot tracking. The existing mobile robot navigated in the task space using developed motion model and the Equations 5.1 to 5.8 in section III. In this case the speed of the mobile robot was kept constant. The rpm of individual Dynamixel EX-106 servo motors was set at 300 (which has full scale value is 2047) and in wheel mode (as the servo motor has two modes i.e. wheel or joint mode). This selection translates to approximately 33 rpm both clockwise and anti-clockwise rotation. The wheels of mobile robot are having 10.2 cm as diameter that achieves 16.6 cm/s linear velocity and 1.107 rad/s angular velocity. To reduce inherent errors in terms of the positional accuracies the EX 106 has been calibrated to get accurate and reliable velocity.

For camera calibration, the positions of the various nodes of the path were mapped from image to pixel coordinates. This process is performed using the transformation matrices obtained during calibration of overhead camera. For calibration of overhead camera MATLAB based calibration routine was used. This routine uses a series of 18 checkerboard images to estimate the camera matrix and distortion parameters. The camera's orientation, with respect to the ground was manually estimated based on the corresponding homographic top view transformation. The raw video resolution was 640×480 at 30 fps. The test cases shown above have been presented in Figures 5.4 and Figure 5.5. For obstacle detection, thresholding, erosion and clustering steps are used. Before thresholding, RGB image is converted to gray image. In this case the global threshold is taken as 200 (which is binary threshold type) in the Open CV software. A series of tests were performed to set a threshold of 10%, so that the detection of an object is considered as success or failure. Using this convention, vision sensor correctly detects the obstacles. Similarly, erosion operator is applied using 3×3 rectangular kernels on the threshold image. Thereafter clustering operator is applied on the image which uses Euclidean distance of 25 pixels.



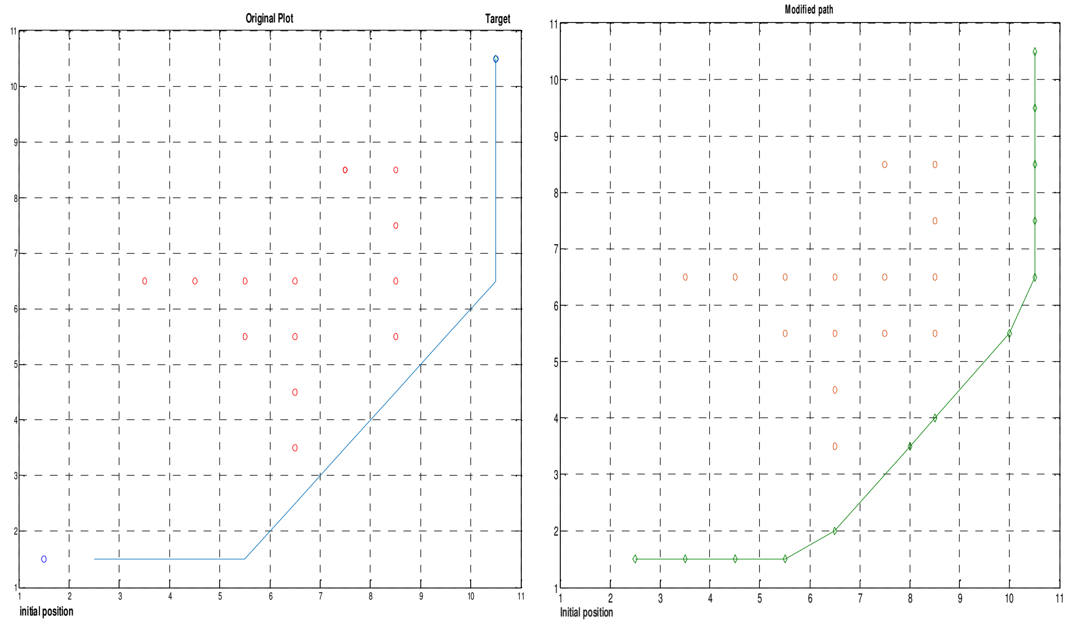
(a) (b) (c) (d)

Figure 5.4: CASE I (a) Captured Image in real-time, (b) Threshold image, (c) Eroded image, (d) Clustered image



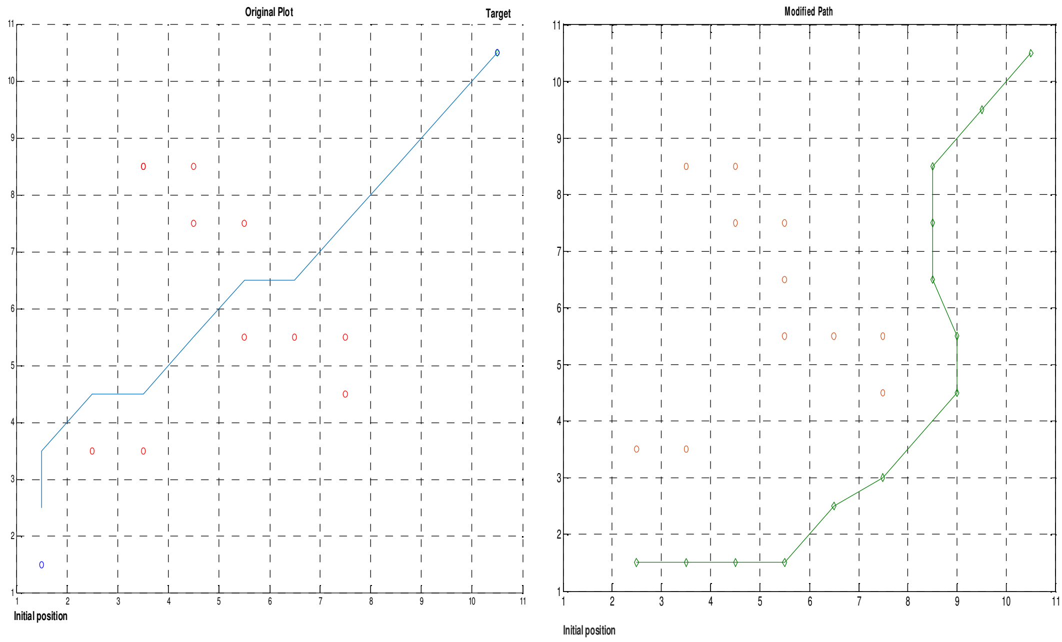
(a) (b) (c) (d)

Figure 5.5: CASE II: (a) Captured Image in real-time, (b) Threshold image, (c) Eroded image, (d) Clustered image



(a) (b)

Figure 5.6: CASE I (a) Original path plot using A*, (b) Modified path plot



(a) (b)

Figure 5.7: CASE II (a) Original path plot using A*, (b) Modified path plot

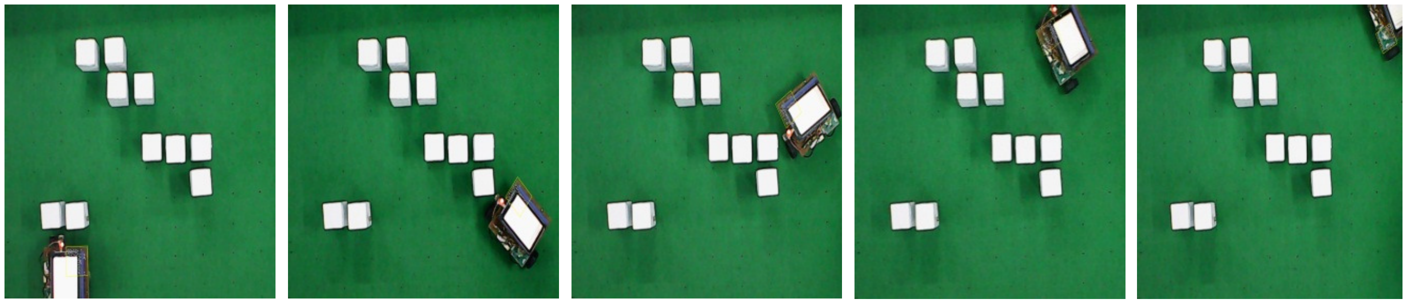
In order to validate the developed shape aware navigation algorithm a simulation was carried out. Based on the image captured in the task space a path for mobile robot is generated using A* algorithm, if distance between two obstacles are less than or equal to 5 cm then the corresponding grid is blocked and alternative path is found out using the shape aware path planning algorithm. If obstacles are so placed that there is no feasible path, then the computer mounted on mobile robot indicate that there is no path for this scenario. For the simulation, any gaps of 5cm between the obstacles are treated as obstacle and an alternative path was found by the algorithm. Here the resolutions of the grids created in the task were set to 12cm and 15cm. Figures 5.6 and 5.7 shows that path planner generates a collision-free path to pass through the narrow passage because it factually considers the shape of the mobile robot.

Here Figure 5.6(a) and Figure 5.7(a) shows the original path using the A* algorithm and Figure 5.6(b) and Figure 5.7(b) shows the modified path (green line) depending on the vision sensor readings. For this algorithm the mobile robot is considered as a 2D object which has a length (L) and breadth (B). The value $M = \max(L, B)$ is defined. Now, if the space (d), between obstacles is greater than M, then path is not blocked. The results of A* and shape aware based A* algorithm is compared. It is tested the obstacle detection and collision-checking algorithm against a narrow passage with A*, as shown in Figure 5.8 and Figure 5.9. Using the above process, the velocity vs. frames plot was found out from the experiments. The velocity refers to the rate of change of displacement of the mobile robot through two consecutive frames. Instead of time, the velocity was found with respect to frames, because the frame rate of the camera was not constant throughout the experiment. The results of the computed *xy*-coordinates of KLT tracking with respect to time and after Kalman filtering with respect to time are explained in result section. Here tracking of mobile robot in a displacement sequence is carried out using KLT algorithm whereas Viola-Jones is used for detecting mobile robot features. For this purpose, 34 videos of the mobile robot backing into many stationary obstacles are captured.

In order to track the mobile robot efficiently a Kalman Filter is used. The Optimal modified path is used as the measurement source for the object's position and is used in the KF's update step. For the prediction step, the motion of the robot and that of the mobile robot was considered. Most of the KF's mathematical details presented here are adapted from same authors (Das et al. 2017).

(a) (b) (c) (d) (e)

Figure 5.8: A* path against a narrow passage of case I



(a) (b) (c) (d) (e)

Figure 5.9: A* path against a narrow passage of case II

The modified path introduces sub-optimality in terms of path planning as the shape of the mobile robot was considered here. The error between the modified path and KLT tracking and KF tracking path were computed at various nodes and the computed error percentage as shown in Figure 5.10(a-c) and Figure 5.11(a-c). For computation of the error Equation 5.14 is used.

$$\text{Percentage of error} = \frac{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\sqrt{U_{\text{map}} \times V_{\text{map}}}} \times 100 \qquad (5.14)$$

where (x_i, y_i) co-ordinates of modified path, (x_j, y_j) co-ordinates of KLT tracking path, $(U_{\text{map}}, V_{\text{map}})$ length and breadth of real time map.

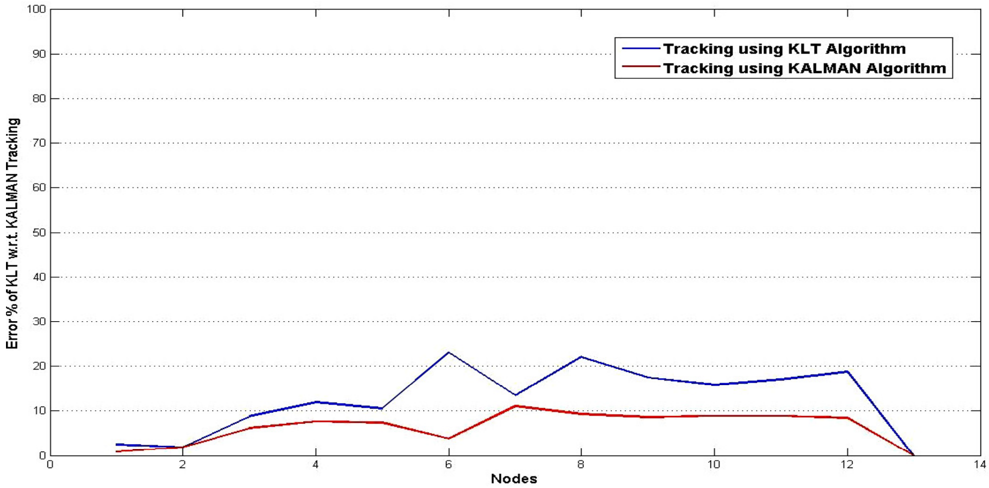
In this exploration every test case is replicated thrice to check the repeatability of the performance. Two different sets of experiments of specified path tracking using KF and KLT based tracking have been analyzed. Percentage error using KLT tracking and KF with respect to frames results in poor tracking and more error. The error is of the order 30 to 40 % due to unsteady velocity and change in the direction of the mobile robot. As per the result it was observed that KF provides better tracking in comparison with KLT tracking.

Table 5.1: Mobile robot orientation with respect to optimal path

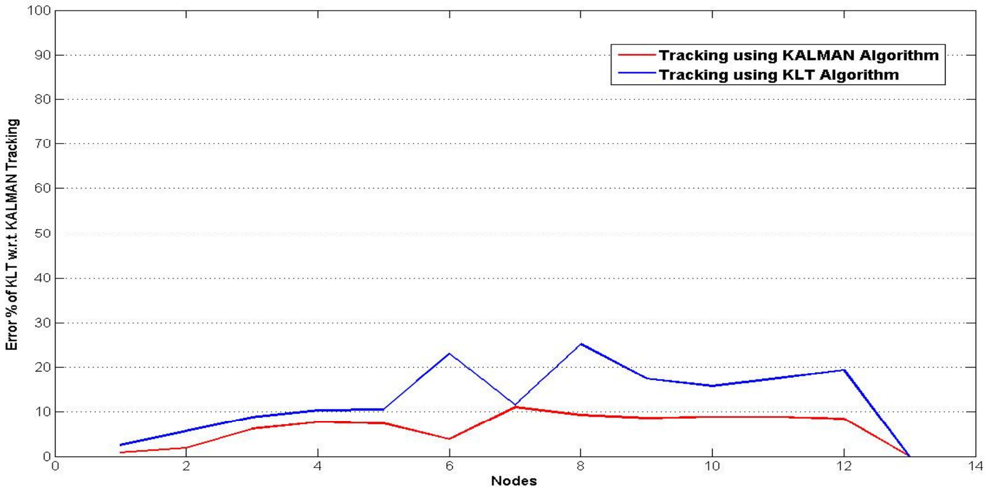
Node	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
(**CASE-I**) Orientation (Degree)	0	0	0	0	13 CCW	0	0	13 CCW	27 CCW	0	0	0	0
(**CASE-II**) Orientation (Degree)	0	0	0	45 CCW	13 CCW	13 CW	45 CCW	27 CCW	27 CW	0	45 CW	0	0

CCW: Counterclockwise, CW: Clockwise

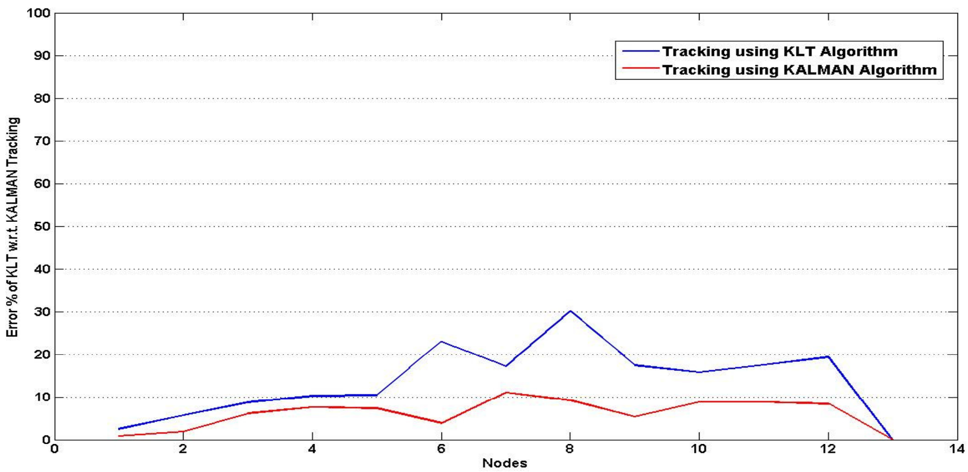
Mobile robot moving toward the target position in modified path with orientation in degree using angular velocity in different nodes is shown in Table 5.1 with cases I and II. In the above table the orientation of mobile robot which moving from start to destination for both cases are provided. For the orientation angle 13 nodes from the task space are taken and the angles in degrees are computed using Equation 5.12 from the kinematic model. In Table 5.1, the mobile robot moves towards the target position in a modified path with orientation in degree shown for different nodes (Node-0 to Node-14). The mobile robot can turn to any degree however; A* algorithm provides a path with either 0 or 45 or 90 degrees with respect to the reference coordinate.



(a)

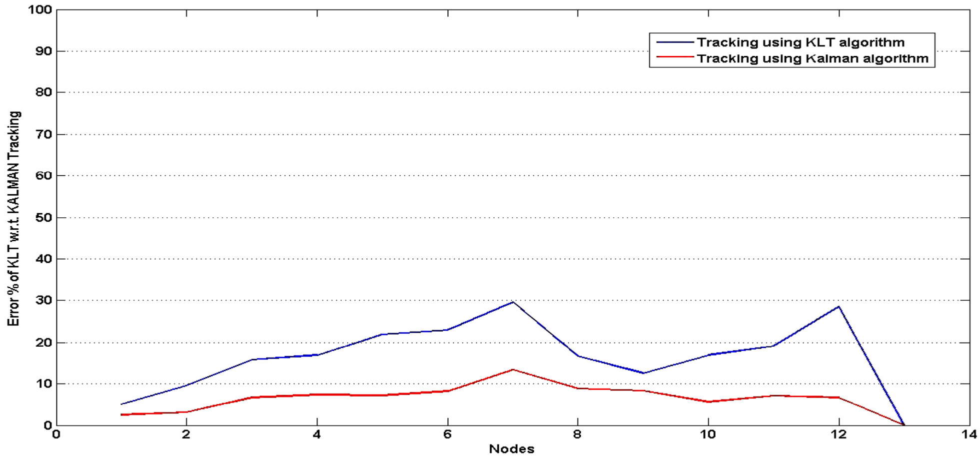


(b)

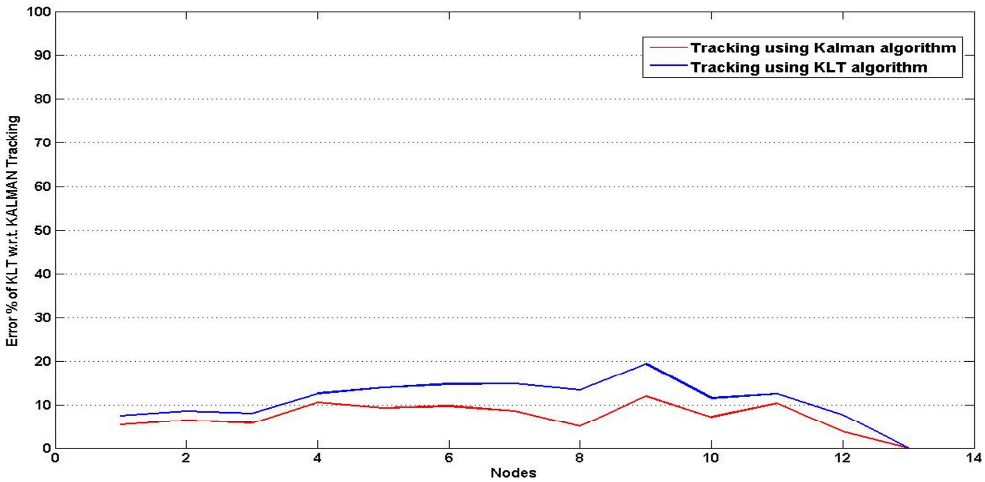


(c)

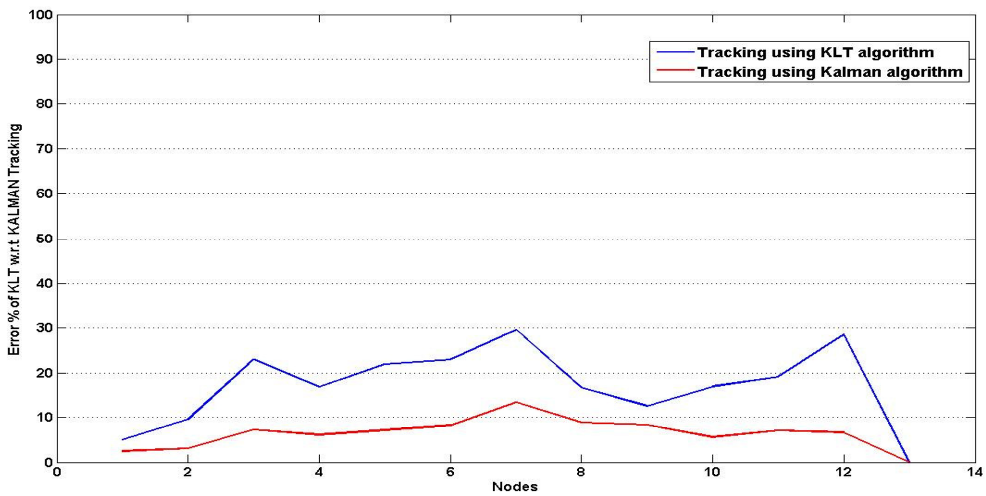
Figure 5.10 (a-c): Percentage of error for KLT tracking and denoising using Kalman filter for test case I



(a)



(b)



(c)

Figure 5.11 (a-c): Percentage of error for KLT tracking and denoising using Kalman filter for test case II

Figures 5.10(a-c) and 5.11(a-c) define the percentage of error for KLT tracking and denoising of the same using Kalman filter for test cases I and II with the help of three trial testing. Current work assumes the complete task space as a grid. The mobile robot determines the path that covers least number of grid cells to reach the destination. A* algorithm uses the heuristic function and computes its value at each node in the task space to get the optimal solution. Nodes are part of the optimal path, and every node is the centroid of the grid cell. The mobile robot passes through these nodes at different instances of time which are task-specific and depend on each other.

Node-0 is the start position of the mobile robot and Node-14 is the target position of the mobile robot. The X-axis of Figures 5.10(a-c) and 5.11(a-c) represent the number of nodes the mobile robot uses to complete the task. The error obtained due to the difference between the simulated position and the tracking position is represented in Y-axis. Here case I and case II represent two tasks under consideration. For both the cases, the angle at every node is specified for the previous node and the rotations specified in clockwise or counterclockwise directions.

5.6 Conclusion

Present work discusses real-time simultaneous shape aware based mapping and navigation algorithm developed for a mobile robot to have collision free path. Obstacle detection has been analyzed with structured real-world environments using vision-based refinement like thresh holding, erosion and clustering. The experiments showed that the proposed modified shape aware path planning algorithm can provide a collision free path with the smallest rotation in the turning direction of the mobile robot. The robot size parameter has been introduced in the proposed heuristic function to determine the navigation of mobile robot. Successful traversal on a planned path was obtained by avoiding narrow passages and maintaining a minimum distance when travelling along a wall or line of objects. The results confirmed that the obstacle avoidance task was successfully carried out while jointly considering shape and kinematics of the robot. Feature point based object detection and tracking method have also been implemented for mobile robot. Tracking of mobile robot has been carried out using KLT algorithm whereas Viola-Jones is used for detecting mobile robot features which are considered for robot navigation. The percentage error between the obtained modified shape aware path with KLT tracking and Kalman filtering were computed to test the efficacy of the proposed algorithm. Here KLT algorithm generates error of the order 30-40% due to unsteady velocity and change in the direction of mobile robot. Kalman filtering generates 6-10% results with respect to KLT tracking. The motivation behind studying this problem is to create a visual surveillance system which has capabilities to track real-time moving object and analyzes its activity. In future, the investigations can be carried out to track mobile robot when the obstacles are also moving in the workspace. Additionally, methods required to improve the overall smoothness of the motion of mobile robot while taking turn should be investigated.