Chapter 8
Vision Based SLAM in Mobile Robots[*]

Abstract. This chapter is an extension of the previous chapter and it discusses how the previously discussed concept of SLAM for mobile robots can be actually implemented in real-life in an indoor environment. The system developed employs a two camera based vision system which successfully performs image feature identification and tracking.

8.1 Introduction

As mentioned in the previous chapter, the extended Kalman filter (EKF) based approach has been widely regarded as probably the most suitable approach for solving the simultaneous localization and mapping (SLAM) problem for mobile robots [1-7]. The basic strength of EKF in solving the SLAM problem lies in its iterative approach of determining the estimation and hence building of an augmented map of its surrounding environment through which the robot is directed to navigate through some waypoints. Here we assume that both the initial localization of the robot pose and the map to be built is unknown to us and we gradually build the map by considering it as an augmentation of estimated states, which are nothing but a collection of the positions of the features or landmarks in the environment, along with the robot's pose. The estimations of these states are integrally associated with some uncertainties in these estimates and they are stored in the form of error covariance matrices. This EKF based SLAM algorithm has been discussed in detail in the previous chapter. In this chapter we shall now discuss how SLAM can be implemented in mobile robots employing vision based sensing.

It is also well regarded that the real implementation of SLAM algorithm for practical environments to build meaningful maps is a difficult task. The accuracy of such a system largely depends on the sensors employed. As we already know, the wheel sensors suffer from wheel-slippage, sonar sensors are low resolution, not highly accurate systems, which also suffer from environmental disturbances,

[*] This chapter is adopted from Expert Systems with Applications, vol. 38, issue 7, July 2011, Avishek Chatterjee, Olive Ray, Amitava Chatterjee, and Anjan Rakshit, "Development of a Real-Life EKF based SLAM System for Mobile Robots employing Vision Sensing," pp. 8266-8274, © 2011, with permission from Elsevier.

infra red sensors can only be employed for short distances, laser range finders are expensive and slow in operation due to low update rate and the performance of GPS can suffer due to occlusion of line-of-sight to satellites and their accuracy and update rate may be slow. Hence, solid-state cameras and computers have emerged in recent times as an attractive, feasible, real-time solution for building such robot localization systems [3, 5]. They can also provide comparatively cheaper solution and they can provide great flexibility in interpreting the environment through which a robotic platform is needed to navigate. However, till date, not many works have been reported utilizing vision sensing based SLAM algorithms. The primary reason for that can be that the development of such systems and to make them meaningfully accurate in real-life is essentially a difficult task.

The present chapter will give a detail description of a successful real-life implementation of SLAM algorithm for map development in an indoor environment [15], utilizing a popular differential drive mobile robot, called KOALA robot, which has also been described in previous chapters. An important highlighting feature of the developed scheme is that this stand-alone system utilizes a computer vision based sensing system for building the map. A two-camera based vision system is utilized to perform feature identification, in frames grabbed, and track these features in subsequent frames. Such a system is essential for scene identification and obstacle recognition for a vision-based system that helps in developing suitable navigational algorithms, performing obstacle avoidance and/or developing a map of the environment where the robot is intended to carry out the navigation job. The feature tracking approach is based on minimization of the sum of squared intensity differences between the past and the current window, which determines whether a current window is a warped version of the past window. The system is also equipped with the 3D distance calculation module of the landmarks from the robot frame, which enables to determine the map of the location, storing current localization of the robot along with the co-ordinates of the landmarks in the map. The system has been implemented in real-life in our laboratory for waypoint-directed map development and the system could demonstrate high accuracy in map development in such indoor environments.

8.2 The Dynamic State Model for the Differential Drive Koala Robot

The details of the EKF based SLAM algorithm were already presented in section 7.2. Now, to adapt this theory in the context of the KOALA robot, at first, the dynamic model is developed for the differential-drive based KOALA robot in this section. This can also be logically extended to other similar types of mobile robots too. Here, there are two independent variables governing motion of the vehicle i.e. rotation of the left wheel of the motor and rotation of the right wheel of the motor. However, we consider two derived variables as primary variables and these are (i) linear translation of the geometric center of the robot and (ii) its rotation around the vertical axis through the geometric center. The rationale behind this domain changeover is because of the reason that an error is introduced if we choose ‘rotation’ as a variable, because of the severe deformation of tier during rotation. Such a problem will not arise in case of linear, translational motion, where the sources of errors or uncertainties are different e.g. incorrect calibration of wheel encoder, small slippage in wheel rotation etc. Here we assume that the robot will never be subjected to simultaneous commands of rotational motion and translational motion.

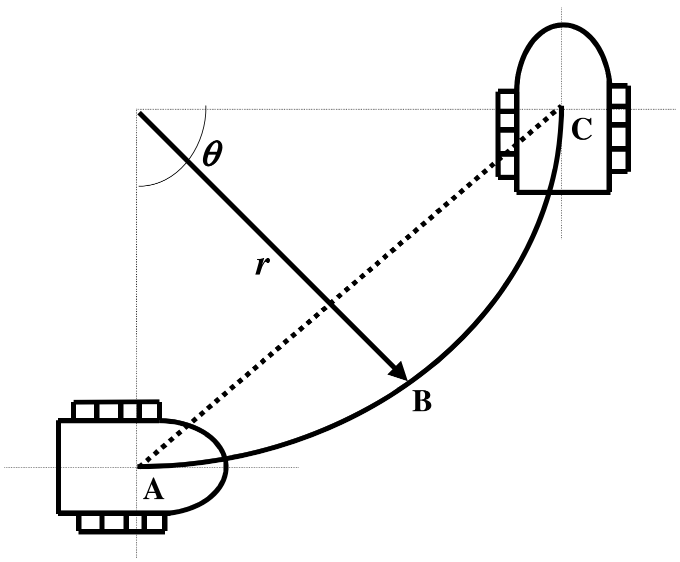


Fig. 8.1. Schematic of the KOALA robot movement

While developing the model, we should keep in mind that the robot always moves along a circular arc. The curvature is zero for linear, translational motion and the radius of curvature is zero for pure rotation. Figure 8.1 shows the schematic of a robot movement. Here

$$\overset{\frown}{A}\overset{\frown}{B}\overset{\frown}{C} = s \tag{8.1}$$

$$\theta = \frac{s}{r} = K_1 \text{ (Rotation of right wheel – Rotation of left wheel)} \tag{8.2}$$

$$s = (K_2/2) \text{ (Rotation of right wheel + Rotation of left wheel)} \tag{8.3}$$

(8.2) and (8.3) enable us to obtain s and θ directly from the readings of the wheel encoders. Hence we obtain, $r = \frac{s}{\theta}$ and $\overline{AC} = 2r\,\mathbf{sin}\,\frac{\theta}{2}$. Then AC can be decomposed into its x- and y-components, when the initial pose ϕ of the robot is known. Therefore we have:

$$\left.\begin{aligned} dx &= 2r\,\mathbf{sin}\frac{\theta}{2}\,\mathbf{cos}\,\phi \\ dy &= 2r\,\mathbf{sin}\frac{\theta}{2}\,\mathbf{sin}\,\phi \\ d\phi &= \theta \end{aligned}\right\} \tag{8.4}$$

The development of such a model gives rise to a logical problem under those situations when $\theta \to 0°$, because then $r = \frac{s}{\theta} \to \infty$. Hence, for $\theta < 5°$, it is assumed that $\overline{AC} = s$. Now, for D amount of linear displacement and θ amount of rotation of the KOALA robot, the dynamic model can be finalized using the following formulae:

$$\left.\begin{aligned} \Delta x &= D\,\mathbf{cos}\phi \\ \Delta y &= D\,\mathbf{sin}\phi \\ \Delta\phi &= \theta \end{aligned}\right\} \Rightarrow \left.\begin{aligned} x_{k+1} &= x_k + D\,\mathbf{cos}\phi \\ y_{k+1} &= y_k + D\,\mathbf{sin}\phi \\ \phi_{k+1} &= \phi_k + \theta \end{aligned}\right\} \tag{8.5}$$

Hence the Jacobians and the covariance matrix will be calculated as:

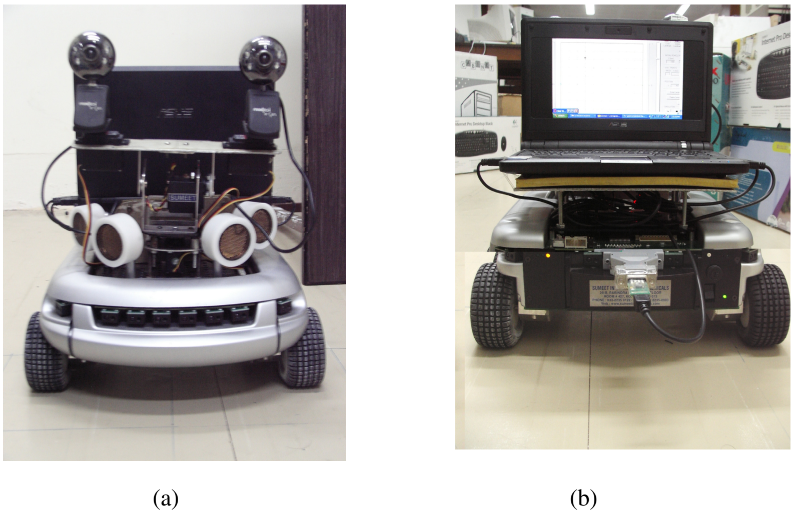
$$\nabla \mathbf{f}_{\mathbf{u}} = \begin{bmatrix} \frac{\partial \Delta x}{\partial D} & \frac{\partial \Delta x}{\partial \theta} \\ \frac{\partial \Delta y}{\partial D} & \frac{\partial \Delta y}{\partial \theta} \\ \frac{\partial \Delta \phi}{\partial D} & \frac{\partial \Delta \phi}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \mathbf{cos}\phi & 0 \\ \mathbf{sin}\phi & 0 \\ 0 & 1 \end{bmatrix} \tag{8.6}$$

$$\nabla \mathbf{f}_{\mathbf{x}_v} = \begin{bmatrix} 1 & 0 & -D\,\mathbf{sin}\phi \\ 0 & 1 & D\,\mathbf{cos}\phi \\ 0 & 0 & 1 \end{bmatrix} \tag{8.7}$$

and $\mathbf{Q} = \begin{bmatrix} \sigma D^2 & 0 \\ 0 & \sigma\theta^2 \end{bmatrix}$ where $\sigma D = D \times$ standard deviation for per unit displacement and $\sigma\theta = \theta \times$ standard deviation for per unit rotation.

8.3 Vision Sensing Based Image Feature Identification, Feature Tracking and 3d Distance Calculation for Each Feature

In our SLAM algorithm, the “observe” step is carried out using vision sensing. The basic version of the KOALA robot is originally procured with some built-in sensors, e.g. incremental wheel encoders and infrared (IR) sensors, and it has been later integrated with several accessories e.g. ultrasonic sensors, wireless radio modem, sensor scanning-tilt-pan system, vision system, servo motors for controlling four degrees of freedom, computing platform etc. All the integrations have been carried out in-house in our laboratory. Figure 8.2 shows the KOALA robot in its integrated form, used specifically for the purpose of performing vision based SLAM.



(a) (b)

Fig. 8.2. KOALA mobile robot, original procured with some built-in sensors, and later integrated in our laboratory with several accessories

The vision-based sensing employs two webcams, as shown in Fig. 8.2(a), for real-life implementation, where the main objective is to implement a two camera based vision system for image feature selection, tracking of the selected features and the calculation of 3D distance of the selected features [16]. This feature identification is based on selection of suitable, candidate image patches or windows in captured frames from running videos acquired from each camera, that have high potential of tracking in subsequent frames. In real life, image patches having high edge information content are better candidates for tracking and hence such patches (considered as static in our system) are considered the best candidate landmarks for developing subsequent maps. The computation of correspondences between features in different views (for our system, the left snap and the right snap i.e. the frames grabbed from the left camera and the right camera) is a necessary precondition to obtain depth information. The system first performs a feature identification algorithm in the frame grabbed from the left camera to identify some suitable rectangular patches or windows that are most suitable as trackable features (patches with sufficient texture) and then it attempts to track them in the frame grabbed from the right camera. The inspiration for developing such a image tracking system is obtained from the Kanade-Lucas-Tomasi (KLT) Tracker [10, 13]. It is always preferable to track a window or patch of image pixels instead of a single pixel because it is almost impossible to track a single pixel, unless it has a very distinctive brightness with respect to all its neighbors. At the same time the result can be confusing, because the intensity value of the pixel can also change due to noise. Hence *N* number of feature windows is selected, based on the intensity profile, by maintaining a minimum distance between the features in an image frame. For an image f(x, y), a two dimensional function, its gradient is a vector and the gradient of each window *G* is calculated along x-direction and y-direction as:

$$G = \begin{bmatrix} g_{xx} & g_{xy} \\ g_{xy} & g_{yy} \end{bmatrix} = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \tag{8.8}$$

The suitability of the choice of a window as a feature window is evaluated by computing the eigenvalues λ_1 and λ_2 of its *G* matrix and a feature window is accepted if

$$\min(\lambda_1, \lambda_2) > \lambda \tag{8.9}$$

where λ is a predefined threshold [14]. Two small eigenvalues mean a roughly constant intensity profile within the window. A large and a small eigenvalue correspond to a unidirectional texture pattern. On the other hand two large eigenvalues represent the corners or salt and pepper type texture [11][16].

Once the features are selected, the next job is to follow or track these features from one frame to another frame in an image sequence [11-13]. Similar to [11], we compute the displacement $\mathbf{dp} = [dxp\ dyp]^T$ of the center of a feature window that minimizes the sum of the squared difference in image intensities between the windows of the two image frames under consideration. In case of the small inter-frame motion, the motion of the features within two image frames can be approximated sufficiently accurately by a pure translation model. However, for bigger inter-frame motions, an affine model, comprising linear warping combined with pure translation, is known to provide better models. Here, the quality of the feature monitored during tracking is better with a dissimilarity measure that includes a deformation matrix that represents the linear warping based affine motion model as well as translations of feature within the frame. The point motion in the image can be described by

$$J(\mathbf{Axp} + \mathbf{dp}) = I(\mathbf{xp}) \tag{8.10}$$

where, J is the current image, I is the original image, $\mathbf{A} = \mathbf{1}+\mathbf{D}$ ($\mathbf{1}$ is a 2x2 identity matrix and $\mathbf{D}$ is the deformation matrix) and $\mathbf{dp}$ is the translation vector. Hence the dissimilarity can be computed utilizing $w(\mathbf{xp})$, a weighting function (popularly chosen as unity or a Gaussian function to emphasize the central portion of the window) as [11]

$$\varepsilon = \iint_W [J(\mathbf{Axp} + \mathbf{dp}) - I(\mathbf{xp})]^2 \, w(\mathbf{xp})\mathrm{d}\mathbf{xp} \tag{8.11}$$

The Newton-Raphson minimization between image intensities of two windows is employed to search for the new position of the center point of a feature window in a new frame in an iterative manner. The following system is needed to be solved to obtain **dp**:

$$\mathbf{Gdp}=\mathbf{e} \tag{8.12}$$

where $\mathbf{G} = \int (\mathbf{gg}^T w)da$; **G** = second order weighted coefficient matrix (2×2), **e** = weighted intensity error vector (2×1) ($\mathbf{e} = (\int_W (I - J)\, \mathbf{g} w da)$, $\mathbf{dp}$ = displacement vector (2×1) ($\mathbf{dp} = [dxp \;\; dyp]^T$), and **g** = Gradient vector (2×1) ($g = \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix}^T$).

This iterative algorithm solves (8.12) by solving, in each iteration, for $\begin{bmatrix} g_{xx} & g_{xy} \\ g_{xy} & g_{yy} \end{bmatrix} \begin{bmatrix} dxp \\ dyp \end{bmatrix} = \begin{bmatrix} e_x \\ e_y \end{bmatrix}$ and calculating the new window center in the image, where we are trying to perform the tracking, in that iteration, as $x_{\text{p\_tracked}} = x_{\text{p\_tracked}} + dxp$; $y_{\text{p\_tracked}} = y_{\text{p\_tracked}} + dyp$.

The 3D distance of the tracked landmarks can be obtained on the basis of data available about the geometry of the camera and the head used [3], [9], [14]. To get depth information in stereo vision, it is required that two lines of sight for the two cameras intersect at a scene point P and from this information the three-dimensional coordinates of the observed scene point in the world co-ordinate system (WCS) can be obtained. Our distance calculation module is based on the pin-hole camera model used in Andrew J. Davison's work [3]. It makes use of the well known camera calibration matrix and perspective projection equation and utilizes the "Midpoint of Closest Approach". Figure 8.3 shows a front view of the active head designed and implemented in our laboratory where **H** = the vertical distance of the head center above the ground plane, **I** = the horizontal distance between the left and the right vergence axes, and c = the offset along either vergence axis between the intersections with the elevation axis and the camera optic axis.

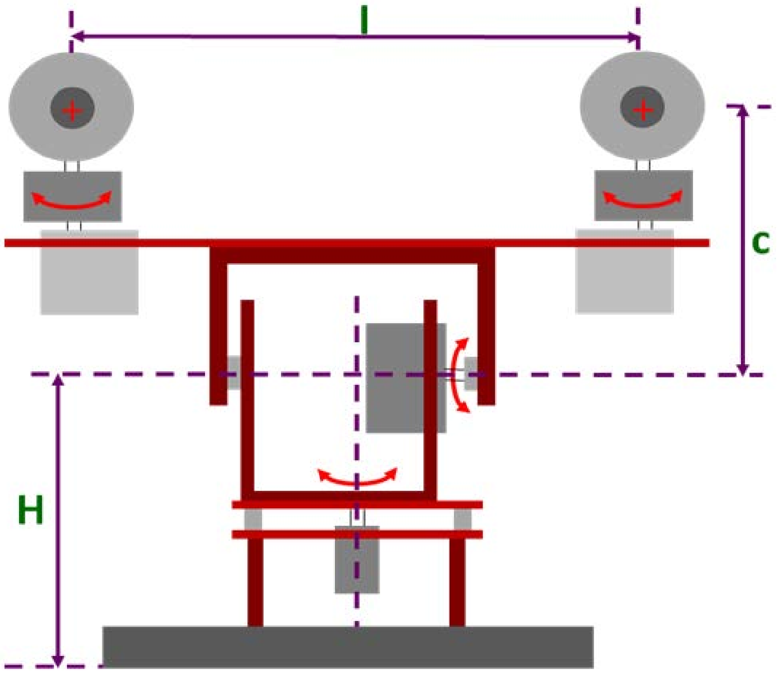


Fig. 8.3. Front view of the active head designed in our laboratory with sensor-scanning-pan-tilt system, two webcams and four servo motors for individual control of four degrees of freedom (pan control, tilt control, left vergence control and right vergence control)

Once new landmarks or image patches are identified and tracked between left and right camera images they can be initialized in the map utilizing the usual procedure of new landmark initialization in our EKF-based SLAM algorithm. Similarly, identification and tracking of image patch(es) in left and right camera images, which was(were) also previously identified in images acquired for a past position of the robot, will constitute the re-observation step of our EKF-based SLAM algorithm. In this step, where the estimated position of this landmark is calculated according to the usual "Predict" step of the Kalman filter, it is further refined by performing the corresponding "Observe and Update" step of the Kalman filter algorithm.

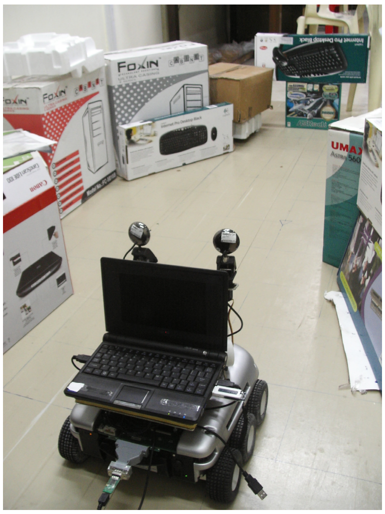
The steps followed for this vision-sensing based real-life implementation of EKF-SLAM algorithm is shown in Algo. 8.1. Here it can be seen that the robot is asked to move through some waypoints and it is directed to build a map of its surrounding. To perform this function, the robot is moved by a specified distance and it grabs several image frames to perform landmark observation as well as its own localization simultaneously. To build a map for both environment ahead of the robot, environment to its left and environment to its right, it is taking image shots both for 0° angular position of the pan-angle, for +θ° angular position of pan-angle and for -θ° angular position of pan-angle. Hence during the "observe" step of the EKF the robot identifies and acquires feature(s)/landmarks(s) from environment straight ahead of it, from environment to its left and from environment to its right. This procedure of moving the robot ahead, performing the "predict" step, using vision sensors in several pan directions to acquire and track landmarks, and to perform "correct and update" step of EKF algorithm is performed in an iterative fashion, until the last waypoint is reached. The map built in the last iteration is utilized as the final map built by the robot, to be used for some future tasks in the same environment.

```
Step 1. Specify the waypoints through which the robot should navigate and
        initialize the robot pose.
Step 2. Move the robot by a specified amount and perform the "predict" step of
        EKF.
Step 3. Grab image frames from continuously running video sequences in left and
        right camera, for 0° angular position of the pan-angle, and perform
        feature identification, tracking and distance calculation of the tracked
        feature(s) from the robot.
Step 4. Repeat Step 3 for +θ° angular position of pan-angle.
Step 5. Repeat Step 3 for -θ° angular position of pan-angle.
Step 6. For new feature(s)/landmark(s) observed in step 3 - step 5, initialize them
        in the map.
Step 7. For those feature(s)/landmark(s) observed in step 3 - step 5, which were
        observed earlier, perform the usual "observe and update" step of EKF, to
        refine the map already built.
Step 8. Perform step 2 – step 7 until the robot reaches the last waypoint specified.
Step 9. Store the last map built by the robot as the final map built for the
        environment.
```

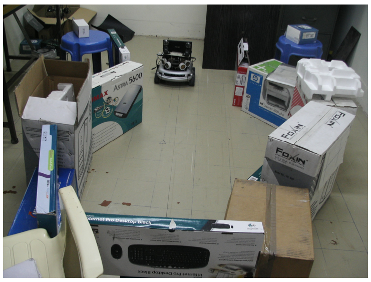
Algo. 8.1. The Real EKF-based SLAM algorithm implemented for the KOALA robot, using vision sensors, in an indoor environment (in our laboratory)

8.4 Real-Life Performance Evaluation

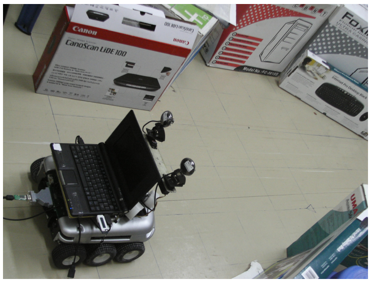
As we have mentioned previously, the KOALA robot is a 32 cm x 32 cm, six wheeled, and differential drive vehicle manufactured by K-team, Switzerland. It has already been mentioned that in KOALA, the hardware control is performed by an on- board microprocessor (16MHz Motorola 68331@ 22MHz) [8]. To add the four degrees of freedom to the robot system for pan, tilt, left vergence and right vergence control, we have developed a PIC 16F876A micro-controller based system that, in interrupt-driven mode, works in conjunction with the Motorola processor of the KOALA robot, in master-slave configuration. The development of such a PIC micro-controller based system for interfacing external add-on peripherals with a real mobile robot, is really helpful for adding flexibility for real life applications and this development was discussed in detail in chapter 2.



(a)



(b)



(c)

Fig. 8.4. The environment created through which the robot navigates and performs EKF-SLAM algorithm

Figure 8.4 shows the indoor environment created through which the robot is asked to navigate through several specified waypoints and build a map performing vision-based SLAM algorithm. To judge the performance of the system, a grid containing 100 squares was drawn on the maze with each square having a dimension of 20 cm × 20 cm i.e. a navigation domain of dimension 2 m × 2 m was explored.

Figure 8.5 shows the GUI-based software developed in our laboratory for real-life execution of the EKF-SLAM algorithm. Different frames in Fig. 8.5 show the landmarks identified during several iterations for incremental map building employing the EKF-SLAM algorithm and incorporation of these landmarks in the stored map. The “green line” shows the ideal path joining the waypoints through which the robot is asked to navigate. The “light blue triangle” represents the initial, starting pose of the robot and, as can be seen in Fig. 8.4, this initial pose for our implementation is considered as: $(z, x, \phi)^T = (-100, 0, 0^\circ)^T$. For this real-life implementation here, the notations z, x and ϕ are chosen in conformation with the notations used in [3] and hence the z-direction and x-direction correspond to the x-direction and y-direction respectively, as specified in our theories before.

During its navigation, the robot identifies landmarks in its surrounding environment and initializes their positions or refines their positions in the map. As the robot keeps moving forward, the number of landmarks identified, and hence, the size of the map, increases. The “red crosses” in the map show the 2D positions of the landmarks identified. Figure 8.5(d) shows the final map constructed at the end of the test-run of the KOALA robot.

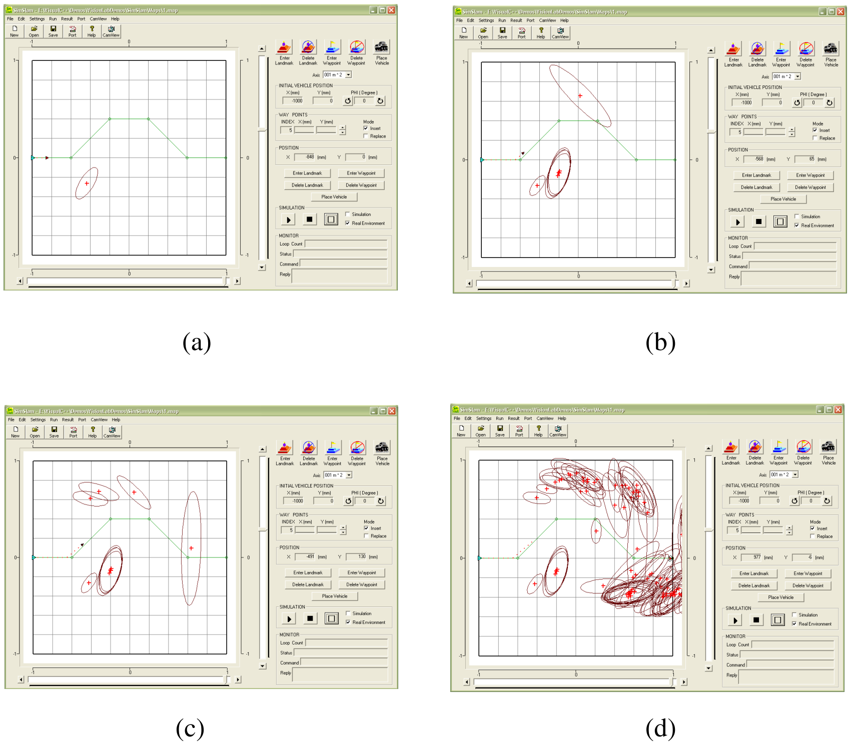
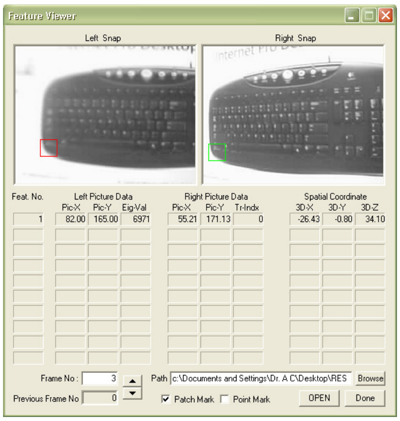


Fig. 8.5. Real-life landmark identification for map building in different steps of EKF-SLAM algorithm

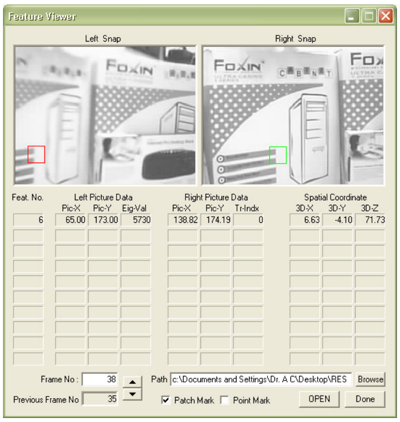
Figure 8.6 shows the GUI-based form developed for capturing image frames in real-life, for some representative positions of the KOALA robot and demonstrating the performance of feature extraction and tracking algorithm, for meaningful identification of landmarks. The image patches identified in “red squares” are identified as new potential landmarks and the image patches identified in “green squares” are identified as re-observed landmarks. The form also displays the 3D distance calculated for each landmark tracked, from the robot.



Feat. No.	Left Picture Data Pic-X	Pic-Y	Eig-Val	Right Picture Data Pic-X	Pic-Y	Tr-Indx	Spatial Coordinate 3D-X	3D-Y	3D-Z
1	82.00	165.00	6971	55.21	171.13	0	-26.43	-0.80	34.10

Frame No : 3
Previous Frame No 0
Path c:\Documents and Settings\Dr. A C\Desktop\RES
☑ Patch Mark ☐ Point Mark

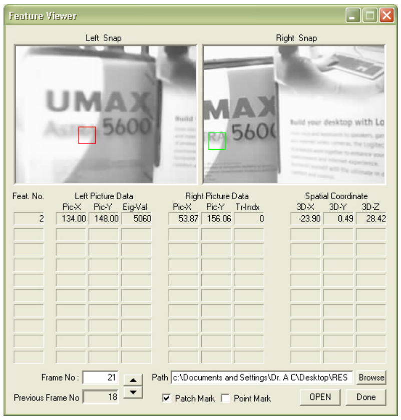
(a)



Feat. No.	Left Picture Data Pic-X	Pic-Y	Eig-Val	Right Picture Data Pic-X	Pic-Y	Tr-Indx	Spatial Coordinate 3D-X	3D-Y	3D-Z
6	65.00	173.00	5730	138.82	174.19	0	6.63	-4.10	71.73

Frame No : 38
Previous Frame No 35
Path c:\Documents and Settings\Dr. A C\Desktop\RES
☑ Patch Mark ☐ Point Mark

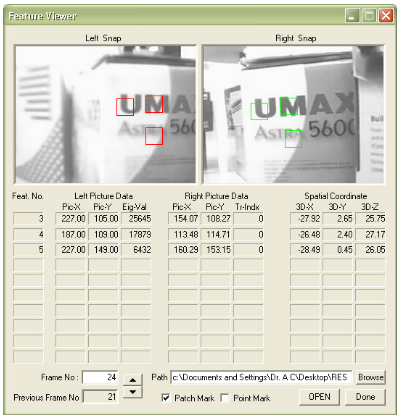
(b)



Feat. No.	Left Picture Data Pic-X	Pic-Y	Eig-Val	Right Picture Data Pic-X	Pic-Y	Tr-Indx	Spatial Coordinate 3D-X	3D-Y	3D-Z
2	134.00	148.00	5060	53.87	156.06	0	-23.90	0.49	28.42

Frame No : 21
Previous Frame No 18
Path c:\Documents and Settings\Dr. A C\Desktop\RES
☑ Patch Mark ☐ Point Mark

(c)



Feat. No.	Left Picture Data Pic-X	Pic-Y	Eig-Val	Right Picture Data Pic-X	Pic-Y	Tr-Indx	Spatial Coordinate 3D-X	3D-Y	3D-Z
3	227.00	105.00	25645	154.07	108.27	0	-27.92	2.65	25.75
4	187.00	109.00	17879	113.48	114.71	0	-26.48	2.40	27.17
5	227.00	149.00	6432	160.29	153.15	0	-28.49	0.45	26.05

Frame No : 24
Previous Frame No 21
Path c:\Documents and Settings\Dr. A C\Desktop\RES
☑ Patch Mark ☐ Point Mark

(d)

Fig. 8.6. Sample examples of results of feature extraction, feature tracking and 3D distance calculation of the tracked features from the robot, for some representative positions of the KOALA robot, during its test run in the environment

Figure 8.7 shows three sample situations of identifying and tracking features/landmarks in real environments. The "green line" on the maze and in vertical direction and the "red dots" help in pointing the actual landmark in the environment and in obtaining its true position. The hollow circle drawn in "light blue" shows the actual object corresponding to an image patch identified in the environment. The estimated positions of these landmarks in the map built, shown earlier in Fig. 8.5, show that there are small discrepancies between the true 2D positions and the estimated 2D positions for most of the landmarks in the map. However this is always understandable and can be appreciated for real-life experimentations. Table 8.1 shows these true and estimated positions, for the three sample landmarks under consideration, as shown in Fig. 8.7.

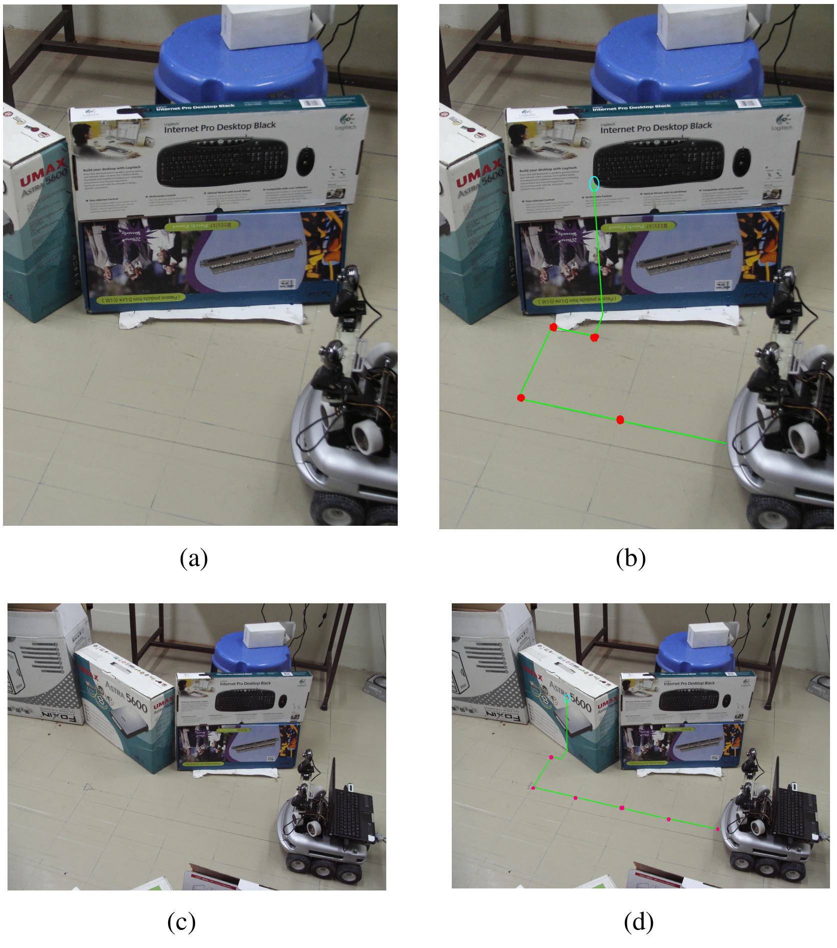
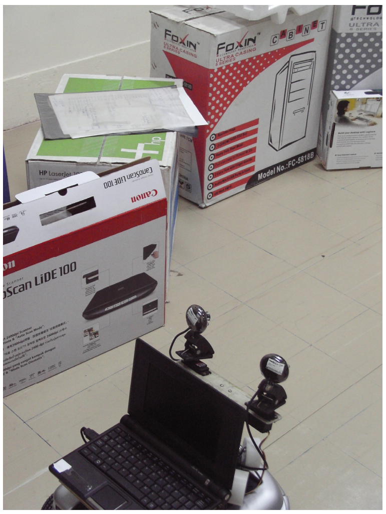


Fig. 8.7. Three sample situations of identifying and tracking landmarks in real environments


(e)

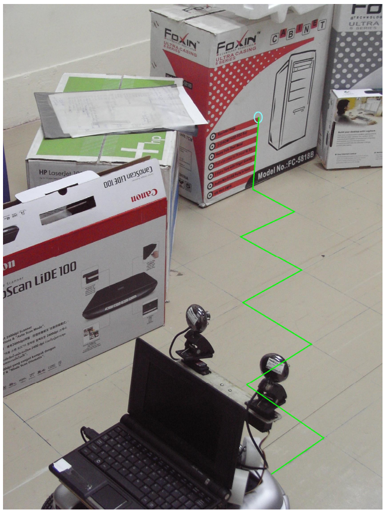

(f)

Fig. 8.7. (*continued*)

Table 8.1. Performance comparison of the EKF-SLAM algorithm employing vision sensing, for three sample real-life landmarks, as shown in Fig. 8.7

Sl. No.	Landmark Description	Estimated Position (cm)		True Position (cm)	
		z-coordinate	x- coordinate	z-coordinate	x- coordinate
1.	Landmark in Fig. 8.6(a) and Fig. 8.6(b) (bottom left corner of the keyboard image)	-43	-26	-47	-27
2.	Landmark in Fig. 8.6(c) and Fig 8.6(d) (corner of the letter 'A' in UMAX box)	-18	-18	-10	-23
3.	Landmark in Fig. 8.6(e) and Fig. 8.6(f) (top right corner of the thick red line in the FOXIN box)	4	70	2	74

8.5 Summary

In this chapter we described the theories of and successfully demonstrated a real-life implementation of the simultaneous localization and mapping problem (SLAM) of mobile robots for indoor environments, utilizing two web-cam based stereo-vision sensing mechanism. The system showed a successful implementation of an algorithm for image feature identification in frames grabbed from continuously running videos on two cameras, installed on the active head integrated in-house with KOALA mobile robot, tracking of features/landmarks identified in a frame in subsequent frames and incorporation of these landmarks in the map created, utilizing a 3D distance calculation module implemented in real-life for calculation of co-ordinates of landmarks in WCS on the basis of the distances calculated of the landmarks from the robot frames. The system could be successfully test-run in laboratory environments where our experimentations showed that there are very small deviations of the estimated landmark positions determined in the map from the actual positions of these landmarks in real-life. It is hoped that such successful implementations will inspire many readers to implement similar meaningful map building systems for more complex environments and also in outdoor situations.

Acknowledgement. The work reported in this chapter was supported by All India Council for Technical Education under RPS scheme (Grant No. 8023/BOR/RPS-89/2006-07).

References

[1] Dissanayake, M.W.M.G., Newman, P., Clark, S., Durrant-Whyte, H.F.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Tran. Robotics and Automation 17(3), 229–241 (2001)
[2] Smith, R., Cheeseman, P.: On the representation and estimation of spatial uncertainty. International Journal of Robotics Research 5(4) (1986)
[3] Davison, A.J.: Mobile Robot Navigation Using Active Vision. PhD Thesis, Univ. of Oxford (1998)
[4] Bailey, T.: Mobile Robot Localization and Mapping in Extensive Outdoor Environments. PhD Thesis, Univ. of Sydney (2002)
[5] Davison, A.J., Murray, D.W.: Simultaneous localization and map-building using active vision. IEEE Tran. Pattern Analysis and Machine Intelligence 24(7), 865–880 (2002)
[6] Chatterjee, A., Matsuno, F.: A neuro-fuzzy assisted extended Kalman filter-based approach for Simultaneous Localization and Mapping (SLAM) problems. IEEE Trans. on Fuzzy Systems 15(5), 984–997 (2007)
[7] Chatterjee, A.: Differential evolution tuned fuzzy supervisor adapted extended Kalman filtering for SLAM problems in mobile robots. Robotica 27(3), 411–423 (2009)
[8] KOALA User Manual, Version 2.0 (silver edition), K-team S.A., Switzerland (2001)
[9] Nishimoto, T., Yamaguchi, J.: Three dimensional measurements using fisheye stereo vision. In: SICE Annual Conference, Japan, pp. 2008–2012 (September 2007)
[10] Brichfield, S.: KLT, An implementation of the Kanade-Lucas-Tomasi feature tracker, `http://www.ces.clemson.edu/~stb/klt`
[11] Shi, J., Tomasi, C.: Good Features to Track. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1994) Seattle, pp. 593–600 (June 1994)
[12] Marr, D., Poggio, T., Ullman, S.: Bandpass channels, zero-crossing, and early visual information processing. Journal of the Optical society of America 69, 914–916 (1979)
[13] Moravec, H.: Obstacle avoidance and navigation in the real world by a seeing robot rover. PhD thesis Stanford University (September 1980)

[14] Yamaguti, N., Oe, S., Terada, K.: A Method of distance measurement by using monocular camera. In: SICE Annual Conference, Japan, pp. 1255–1260 (July 1997)
[15] Chatterjee, A., Ray, O., Chatterjee, A., Rakshit, A.: Development of a Real-Life EKF based SLAM System for Mobile Robots employing Vision Sensing. Expert Systems with Applications 38(7), 8266–8274 (2011)
[16] Chatterjee, A., Singh, N.N., Ray, O., Chatterjee, A., Rakshit, A.: A two-camera based vision system for image feature identification, feature tracking and distance measurement by a mobile robot. International Journal of Intelligent Defence Support Systems 4(4), 351–367 (2011)