EKF and K-means to Generate Optimized Paths of a Mobile Robot

Md Nasir Uddin Laskar, Hoang Huu Viet, TaeChoong Chung
Artificial Intelligence Lab, Dept. of Computer Engineering
Kyung Hee University, 446-701, South Korea
{nasir, viethh, tcchung}@khu.ac.kr

Abstract

Finding optimized path in the workspace is one of the fundamental problems to solve for an autonomous mobile robot. Avoiding obstacles and building an efficient trajectory is the key goal. For this reason, a mobile robot has to manage the free configuration space very efficiently for the purpose of path planning and navigation. Partitioning the configuration space will make the path planning task easy, faster and efficient. Also, data read by the sensor has some inherent noise. So we implement an algorithm to make an efficient estimation of the future states to build map that helps manage the environment efficiently to find the optimized paths to destination. We apply Extended Kalman Filter (EKF) to find the accurate estimation on sensor data and then K-means clustering algorithm to find the next landmarks avoiding the obstacles.

Keywords: *Extended Kalman Filter (EKF), K-means clustering, configuration space, path planning, robot navigation.*

1: Introduction

Path planning is arguably the most important goal for a robot to solve. It deals with automatic generation of feasible paths for robots in the presence of obstacles. So far, the major concern was to merely find a feasible path, regardless of its quality. But these days, quality of the path carries more interest than merely finding a collision free path. This problem has been solved mostly by using randomized algorithms, such as Probabilistic Road-Maps (PRM) [1] or Rapidly-exploring Random Trees (RRT) [2]. Sampling-based path planning technique [3] handles complex problems in high-dimensional spaces but usually operate in a binary world aiming to find out collision free solutions rather than the optimal path. Classical grid-based methods [4] can be used to compute resolution-optimal paths over a costmap. To find paths quickly in large search spaces, roadmap based planners are ideal [5].

Optimization of generated paths using randomized algorithms is addressed by [6] that presents an iterative algorithm to optimize raw paths. Some of the methods extract optimized paths from motion planning roadmaps [7] enabling collision detection and kinematic constraints. This readmap is created by using reachability roadmap method which is particularly suited for motion planning in virtual environments. For robotic applications, a path should be short and should keep some amount of minimum clearance to the obstacles. Such a path is generated using partial shortcut [8] technique that reduces the chances of collisions. Considering a cost function defined over the configuration space, L. Jaillet *et. al.* [9] proposes a low-cost Transition-based RRT planner that finds the solution with respect to a path quality criterion.

Some robots with inexpensive sensors rely on heuristics [10] or randomized approach because of the technical difficulties of implementing localization, map building and path planning with low-cost sensors as well as low computing power. Many coverage path planners either implicitly or explicitly use cellular decomposition [11] of the free space to achieve some guaranteed level of coverage. They divide the entire region into cells and each cell is covered one after another, which is found to be more efficient than covering the entire region as a whole [12].

To navigate and to find the optimized path in complex environments, autonomous robots require map information from robot sensors. Lumelsky *et al.* [13] considered the sensor-based approach, which requires a robot to build a complete map and cover the whole terrain in an unknown environment based on the sensory feedback. Vision-guided range detection sensors can be used to obtain the floor map [14]. A complete coverage navigation algorithm in unknown environments is proposed by Park and Lee [15] that is composed of a sweeping algorithm, a point-to-point moving algorithm, and a corner work algorithm. Butler *et al.* [16] suggested a distributed coverage algorithm, where the robot can detect obstacles by only sensing in a shared, connected rectilinear environment. In this method, a coverage path is planned in an initially unknown environment by collecting the incoming sensor data. Some localization approach use neural networks for robot map building and map learning [17]. Yasutomi *et al.* [18] proposed a learning-based complete coverage approach using grid-like map representation of workspace in an unknown environment.

The problem of building a grid map using cheap sonar sensors is addressed by [19]. It tries to model the environment as accurately as possible by using the erroneous sensors. Some papers also focus on the map building of multiple robots. Lourades *et al.* [20] considered a team of robots with different sensing and motion capabilities. Chaoxia *et al.* [21] represented a topological map and an online map construction approach. A neural-dynamic based approach for real-time map building and complete coverage navigation of autonomous mobile robots is introduced in [22]. EKF based localization to build the large scale indoor maps is used in [23].

However there are always errors associated with every model. Moreover, sensors are prone to errors and generate noise which adds to the complexity of the problem [24]. So prior processing of the sensor data is essential to generate accurate maps. That is why we first apply EKF to refine the sensor data and then K-means clustering algorithm to find the estimation as accurate as possible.

The remainder of this paper is organized as follows: section 2, gives some idea about the kinematics of the mobile robots with its workspace and configuration space. Section 3, provides an overview of Extended Kalman Filter (EKF) technique. Section 4, introduces the idea of K-means clustering algorithm which is also a relevant component for our method. Section 5 and 6 give our method and the experimental results respectively. Finally we conclude with section 7.

2: Workspace and Configuration Space

The robot can be represented as a point (x, y) E R^2 in the continuous Cartesian plane. The point (x, y) fully describes the coordinate position of the robot in the plane. Nevertheless, this representation is not sufficient for a rigid robot which is capable of translating and rotating in the plane. In this work, the orientation of the robot at each position need be considered and the *configuration* of the robot is defined as

$$q = [x\ y\ 0]^T, \tag{1}$$

where (x, y) is the coordinate position of the robot's center in the Cartesian plane, and 0 is its heading angle.

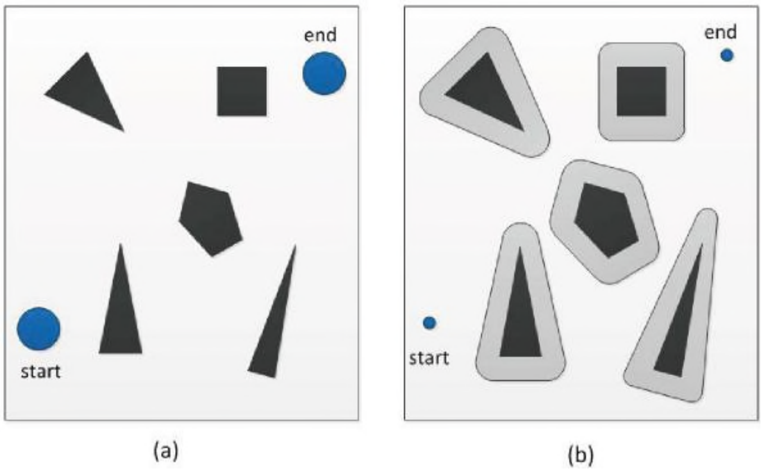

Figure 1. Workspace and configuration space: (a)workspace (b)configuration space

When the robot A moves on a workspace W E R^2, not all regions of W are accessible to it. Since the configuration q of robot A is a specification of the physical state of A with respect to a fixed position in W, a configuration q of robot A is said to be valid if robot A can access the position of W as (2).

$$C = \{\ \forall q\ |A(q)\ \subset W\} \tag{2}$$

The set of all valid configurations of robot A is defined as its *free space*, C_{free}. Configuration q in W that robot A cannot access is called invalid, or it corresponds to *obstacles*. In other words, the configuration space of robot A is the space C of all possible configurations of A in the workspace W. Let O_i be an obstacle in W, then the union of all O_i becomes the obstacle region in W. It is the set of all configurations of robot A at which the robot region A(q) is in contact or overlapping with obstacles regions O_i as (3).

$$C_{obs} = \{\ \forall q, i|A(q)\ n\ O_i \neq \emptyset\} \tag{3}$$

Since the obstacle region prohibits certain configurations of the robot, the free space C_{free} of the robot is defined as

$$C_{free} = \{q\ E\ C|A(q)\ n\ (\bigcup_i O_i) = \emptyset\}. \tag{4}$$

Where $A(q)$ is the portion of W occupied by robot A when the robot is in configuration q. A polygonal or circular robot in the workspace W is represented by a point in configuration space C as Fig. 1, and any point in configuration space corresponds to some placement of an actual robot in workspace.

A path for the robot maps to a curve in the configuration space C and every placement along the path maps to the corresponding point in configuration space. A collision-free path maps to a curve in the free space C_{free}. Fig. 1 illustrates this for a translating circular robot. On the left, the work space W is shown where black colored polygons are the obstacles. Configuration space C is shown on the right side where grey parts are offset areas of the obstacles. The unshaded part in between the grey area is the free space, C_{free}. For clarity, the obstacles are still shown in the configuration space, although they have no meaning there.

3: Extended Kalman Filter (EKF)

Our real world systems are nonlinear. So we apply EKF algorithm to linearize the inputs to get the estimate of the state $x \in \mathbb{R}^n$, allowing nonlinear systems to be modeled. We can analyze [25] the estimation around the current nonlinear kinematics and observation models using the Taylor expansion of the process and measurement function to compute the future estimates.

The EKF, consists of three steps that work in a cycle. First, the robot state and covariance are predicted (EKF prediction). Then, it observes the environment and if there is a new observation or measurement, this is assigned to the correct feature (EKF observation) and the robot state and covariance are finally corrected (EKF update) [26].

Let us assume that our process has state vector $x \in \mathbb{R}^n$, the process is governed by the following non-linear stochastic difference equation

$$X(t+1|t) = f(X(t), U(t), w(t)), \tag{5}$$

where $X(t+1|t)$ is the state prediction based on the current time t with a measurement $z \in \mathbb{R}^m$

$$z(t+1) = h(X(t+1), v(t+1)), \tag{6}$$

here $w(t)$ and $v(t)$ represent the process and measurement noise respectively. These are assumed to be independent and white with normal probability distribution as (7) and (8).

$$p(w) \sim N(0, Q) \tag{7}$$

$$p(v) \sim N(0, R). \tag{8}$$

In practice, one does not know the individual values of the noise $w(t)$ and $v(t)$ at each time step. However, we can approximate the state and measurement vector without them as

$$\hat{X}(t+1|t) = f(\hat{X}(t), U(t), 0) \tag{9}$$

and

$$\hat{z}(t+1) = h(\hat{X}(t+1), 0)) \tag{10}$$

where $\hat{X}(t + 1)$ is an a posteriori estimate of the state from the current time step t. The covariance prediction is

$$P(t+1|t) = \nabla f\Xi(\tau)\ P(t|t)\ \nabla f^{T}\Xi(\tau) + W(t)\,Q(T|T)\,W^{T}(t) \tag{11}$$

where $\nabla f\Xi(\tau)$ is the dynamics gradient (Jacobian) used in order to estimate state $\hat{X}$ and $W(t)$ is the Jacobian matrix of partial derivatives of f with respect to w.

$$\nabla f\Xi(\tau) = \frac{\partial f}{\partial X}, W = \frac{\partial f}{\partial w} \tag{12}$$

The predicted covariance depends on the dynamics and previous covariance $\nabla f\Xi(\tau)\ P(t|t)\ \nabla f^{T}\Xi(\tau)$. The second term in The second term in (11) increases the covariance and the uncertainty until an observation arrives.

The state update $\hat{X}(t + 1|t + 1)$ can be found by (13)

$$\hat{X}(t+1|t+1) = \hat{X}(t+1|t) + K(t+1)\,[z(t+1) - \hat{z}(t+1)] \tag{13}$$

where $z(t + 1)$ and $\hat{z}(t + 1)$ comes form (6) and (10) respectively and the predicted Kalman gain $K(t + 1)$ is found as (14).

$$K(t+1) = P(t+1|t)H^{T}(t+1)\,S^{-1}(t+1) \tag{14}$$

where $S^{-1}(t + 1)$ is the inverse of the innovation covariance and calculated according to

$$S(t+1) = H(t+1)\,P(t+1|t)\,H^{T}(t+1) + V(t+1)\,R(t+1|t)\,V^{T}(t+1) \tag{15}$$

where, H and V matrices are also Jacobian matrix of partial derivatives of h with respect to x and v respectively as (16).

$$H = \frac{\partial h}{\partial X},\ V = \frac{\partial h}{\partial v} \tag{16}$$

The state covariance matrix is updated by

$$P(t+1|t+1) = [I - K(t+1)\,H(t+1)]\,P(t+1|t). \tag{17}$$

Finally EKF algorithm outputs are state updates from (13) and covariance updates from (17). And this iterative process goes on and on.

4: K-means Clustering Algorithm

Given n observations, *K-means* clustering [27] is a method of cluster analysis that partitions n observations into k clusters in the way each observation belongs to the cluster with the nearest mean. Let n positions ($X_1, X_2, \ldots, X_v$) of the robot A where X_i's are consists of 2-D position (x_i, y_i) and orientation 0. K-means clustering is intended to partition the n points into k sets ($k \leq n$) so as to minimize the sum of squares within the clusters according to

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{X_i=k}\sum_{X_j=k}\|X_i - X_j\|^2 = \sum_{k=1}^{K} n_k \sum_{X_i=k}\|X_i - \mu_k\|^2, \tag{18}$$

Algorithm 1: K-means clustering

```
    input : a set of points {$X_1, X_2, \ldots, X_\nu$}
    output: partition into $k$ sets $C = \{C_1, C_2, \ldots, C_\kappa\}$
1  begin
2  |   Find an initial set of $k$ means $\mu^1_1, \mu^1_2, \ldots, \mu^1_\kappa$
3  |   for $i = 1$ to $n$ do
4  |   |   for $j = 1$ to $k$ do
5  |   |       $C^{(\tau)}_\iota = \{x_\pi : \|x_\pi - \mu^{(\tau)}_\iota\| \le \|x_\pi - \mu^{(\tau)}_\varphi\|\}$
6  |   |   end
7  |   end
8  |   find new means based on (19)
9  |   repeat steps 3 and 4, till no point switches clusters
10 |   return cluster centers $\{C_1, C_2, \ldots, C_\kappa\}$
11 end
```

where μ_κ is the mean vector for the $k^{\tau\eta}$ cluster, C_ι is cluster number for the $i^{\tau\eta}$ point within the cluster $W(C)$. For a given cluster assignment C_ι of the data points, compute the new cluster mean $\mu^{(\tau+1)}_\iota$ according to

$$\mu^{(\tau+1)}_\iota = \frac{1}{|C^{(\tau)}_\iota|} \sum_{\Xi j \in X^{(t)}_i} X_\varphi. \qquad (19)$$

In our algorithm, these cluster centers will help building the global map of the environment. For the positions of ($X_1, X_2, \ldots, X_\nu$), we exclude the positions for the obstacles. $\{C_1, C_2, \ldots, C_\kappa\}$ will provide us the k-centers only for the free area in the environment.

In Algorithm 1, widely used initialization methods are Forgy and Random partition. The Forgy method chooses k observations randomly from the data set and uses these as the initial means. On the other hand, the Random partition method assigns a cluster to each observation randomly first and then proceeds to the update step, thus computing the initial means to be the centroid of the cluster's randomly assigned points. So K-means clustering partitions the unknown environment into k clusters by least-squares partitioning method that divides a collection of objects into k disjoint clusters by iteratively minimizing the within-cluster sums-of-squares.

5: Proposed Method

Exploring the free space and finding the best path by building a good map representation is a challenging mobile robot problem. Here we propose a method by which the robot explores its environment without any prior information about it. If the robot knows some destinations throughout the accessible area $C_{\phi\rho\varepsilon\varepsilon}$ in the configuration space C, then it becomes easy for the robot to localize itself and find the optimized path in the environment. To accomplish this, we develop a clustering technique that partitions the robot's free space $C_{\phi\rho\varepsilon\varepsilon}$ in a number of regions, the clusters. Under certain assumptions, the cluster centers can be regarded as the next landmark to be localized.

Algorithm 2: Finding future states

```
input : initial readings from the environment
   output
   EKF({$X_1$, $X_2$ , .... , $X_n$})
1  for
2  begin   {$C_1$ , $C_2$ , .... , $C_k$} centers of the partitions
3          $i$ = 1 $to n$ do
4           $U_x$   $X_{min}$   ($X_{max} - X_{min}$)
5           $U_y$   $Y_{min}$   ($Y_{max} - Y_{min}$)
6       end
7       (for $m$ obstacles)
8       for $i$ = 1 $to$ $m$ do
9           $6_{x1}$   $min(X)$
10          $6_{y1}$   $min(Y)$
11          $6_{x2}$   $max(X)$
12          $6_{y2}$   $max(Y)$
13      end
14      (track indices of forbidden configurations)
15      for $i$ = 1 $to n$ do
16          if ($U_x$  $a_{x1}$ && $U_x$  $a_{x2}$) then
17              if ($U_y$  $a_{y1}$ && $U_y$  $a_{y2}$) then
18              $ind_{obs}$        $i$
19              end
20          end
21      end
22      (update $X$ and $Y$ )
23      for $i$ = 1 $to$ $k$ do
24          $U_x$    $U_x - ind_{obs}(x)$
25          $U_y$    $U_y - ind_{obs}(y)$
26      end
27      K-mean($U_x$, $U_y$)
28      return {$C_1$ , $C_2$ , .... , $C_k$}
29 end
```

Regions of unknown cells completely surrounded by occupied cells or obstacles C_{obs} are removed from the clustering process prior to applying K-means, as they are not reachable. Afterwards, the contour of every region is tracked for determining if it is totally bounded by occupied cells. In that case, the whole region is marked as occupied. First take the sensor data and apply EKF on it. EKF provides us the corrected measurements of the environment which are more accurate than the sensor readings. We apply K-means clustering algorithm on the result found from EKF. K-means clustering gives us k cluster centers by avoiding the obstacles. As the cluster centers are created avoiding the obstacles, these cluster centers can be used for localizing and map building of the robot for this environment as a whole. After the localization is done, it becomes easy for the robot to find the optimized paths to the destination.

Input to Algorithm 2 is the initial data reading from the environment through it’s sensing devices. The world is non linear and sensors have some noise on them. So we apply EKF first to linearize the initial reading. U_x and $U,$ are the coordinates of x and y respectively for the uniform random points in the range. The number n of these uniform random points is a parameter to get the degree of beauty of the center points. The more the random number we generate the more excellent result we achieve. It is also obvious that, the running time of the algorithm depends on choosing this number, as an exceedingly big n value will slow down the algorithm. For our case here, n = 1000 works just fine. $\square_x$ and $\square,$ are the positions for the obstacles. We mark them as forbidden regions by keeping track of $_{indobs.}$ We update U_x and $U,$ by excluding the values of $\square_x$ and ,. Now we have the positions of the free configuration space $Cfree$ only where the robot is allowed to go. So we apply K-means clustering algorithm to find the next candidate landmarks on $Cfree$.

6: Experimental Results

The implementation of the suggested method is done in MATLAB on several environments with random position of obstacles.

6.1: Environment with non-linear Boundary

We consider the value of k in K-means clustering is 12. However, this value is dependent on the range of the sensor. Value of k is proportional to the station of the robot for the range. A greater k value will perform better in an environment which is very cluttered by the obstacles. In Figure 2, where $Cobs$ is more than 21% of C, we tested our method.

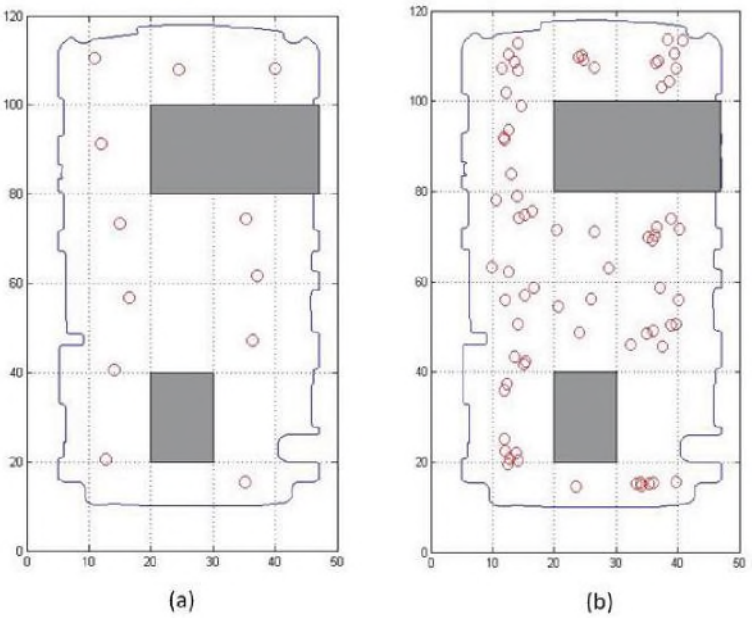

Figure 2. Results for the simulation 1: (a)initial execution (b)three consecutive execution for better understanding

Starting with the values of $Xmin$, $Xmax$, $Ymin$ ^{and} $_{Ymax}$ with 5.5001, 46.6007, 10.5481 and 117.5000 respectively. We generate 1000 uniform random points. For ($\square_x$, $\square,$), we got 16

Table 1. statistics of the simulation for fig. 1(b)

simulation	value of k	no. of forbidden points, $(5_\xi, 5_\psi)$	valid points,(U_ξ, Uy)
1	12	169	831
2		165	835
3		171	829

points that are resided inside the forbidden regions $C_{o\beta\sigma}$, so we exclude them and get 984 permissible points for the K-means clustering algorithm. After applying K-means clustering algorithm on 984 points for k = 12, we get the result of Fig. 2(a).

This simulation is executed on the same environment with consecutively 3 times in a row to get the better understanding of our method. The result we get in this case is quite impressive. Our algorithm can partition the environment and finds the center of each k partitions and successfully excludes the regions that hold the obstacles. We start with the same values for $X_{\mu\iota\nu}$ = 5.5001, $X_{\mu\alpha\xi}$ = 46.6007, $Y_{\mu\iota\nu}$ = 10.5481 and $Y_{\mu\alpha\xi}$ = 117.5000 and generate 1000 uniform random points. For5_ξ and5_ψ we got the points that are resided inside the obstacle regions as in Table 1. $(5_\xi, 5_\psi)$ are the points less than the maximum X (and Y) value and greater than the minimum X (and Y) value of obstacle regions. As we generate the values of U_ξ and U_ψ randomly, number of forbidden configurations $(5_\xi, 5_\psi)$ also vary accordingly. So we exclude them and find all valid points for the K-means clustering algorithm. After applying K-means clustering algorithm on only the valid configurations for k = 12, we come up with the result of Fig. 2(b).

We also see from Table 1 that, the number of allowed configurations (U_ξ, U_ψ) E $C_{\phi\rho\varepsilon\varepsilon}$ is some what stable. The drastic change of the number of (U_ξ, U_ψ) points will imply the instability of the method. So, we see our method is quite stable to produce the cluster centers and ensures the efficient partition of the environment.

6.2: Environment with Large Set of Obstacles

Consider the configuration space described in section 2. For k = 14, we come up with Fig. 3(a) that shows the predicted landmarks. Predicted future states are circled red and robot start and end positions are circled blue as before. A robot is considered as a point in it's configuration space. Shape of the obstacles in Workspace W are colored black and the grayed parts come after the offsetting of the obstacles to construct the configuration space C. Now, $C_{o\beta\sigma}$ is the grayed portions along with obstacle shapes.

In this environment, around 50% of the whole region is surrounded by obstacles $C_{o\beta\sigma}$. After successfully excluding $C_{o\beta\sigma}$ from C, K-means algorithm is applied to $C_{\phi\rho\varepsilon\varepsilon}$ and the result is Fig. 3(a). Our method finds the landmarks for $C_{\phi\rho\varepsilon\varepsilon}$ which can be used to generate the optimized path to the destination. Some of such paths are shown in Fig. 3(b). In some points, it seems that, path of the robot A is touching the obstacle boundaries. This is because of configuration space. Though the robot A is considered as a point in it's configuration space, we showed it as a small circle for the sake of understanding. Moreover, it is allowable for the robot to touch the obstacle boundaries in it's configuration space.

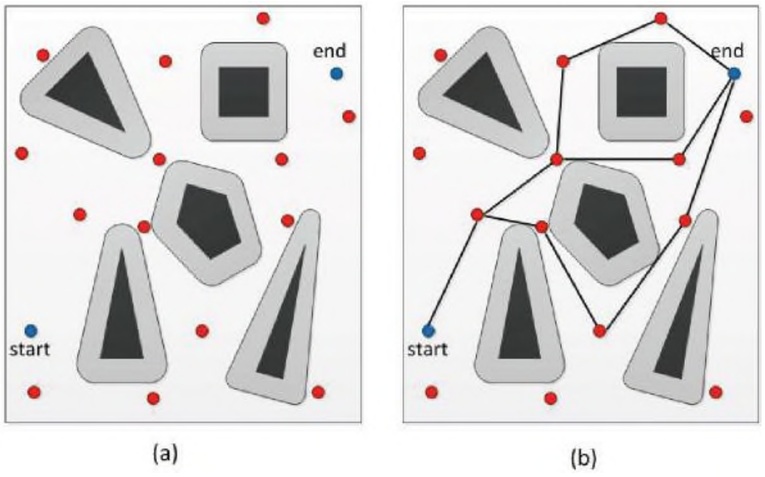

Figure 3. Simulation for the configuration space (predicted states are circled red): (a)predicted landmarks for $k = 14$ (b)example optimized paths.

7: Conclusions

An integrated Extended Kalman Filter (EKF) and K-means clustering algorithm for optimized path generation is proposed. Finding the best path is the key interest for a mobile robot to perform its designated task. These days, autonomous mobile robots are used in more and more complex environments. So the need for tractable ways of planning under uncertainty is growing faster. A robot is not sure about the exact consequence of executing a certain action and its sensor observations are noisy. We consider the noise and linearized it with EKF and finally apply K-means clustering algorithm to find the cluster centers in the environment avoiding the obstacles. Centers of the clusters can be regarded as the next landmarks for the robot to localize itself, and afterwards, these landmarks will be used to find the optimum path to the goal.

Acknowledgements

This work was supported a grant from the NIPA (National IT Industry Promotion Agency) in 2012 (Global IT Talents Program). This research was also partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (2010-0012609).

References

[1] L.E. Kavraki, P. Svestka, J. C. Latombe, M.H. Overmars, ”Probabilistic road-maps for path planning in high-dimensional configuration spaces,” *IEEE Transactions on Robotics and Automation* 12 (4) (1996) 566-580.

[2] S.M. Lavalle, *Rapidly-Exploring Random Trees: A New Tool for Path Planning*, Tech. Rep. 98-11, Computer Science Department, Iowa State University, 1998.

[3] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principle of Robot Motion: Theory, Algorithms, and Implementations*, Cambridge, MA: MIT Press, 2005.

[4] A. Stentz, ”Optimal and efficient path planning for partially-known environments,” Proc. IEEE Int. Conf. on Robotics and Automation, pp. 3310-3317, 1994.

[5] L. Kavraki, M. Kolountzakis, and J. C. Latombe. ”Analysis of probabilistic roadmaps for path planning,” *IEEE Trans. Robot. Automat.*, volume 14, pages 166-171, 1998.

[6] R. Guernane and N. Achour, ”Generating optimized paths for motion planning,” *Robotics and Autonomous Systems*, vol. 59, pp. 789-800, 2011.

[7] J. Kim, R. A. Pearce and N. M. Amato, ”Extracting optimal paths from roadmaps for motion planning,” in *proc. of Int. Conf. on Robotics and Automation*, 2003.

[8] R. Geraerts and M. H. Overmars, ”Creating high-quality paths for motion planning,” *The Int. Jour. of Robotics Research*, vol. 26, no. 8, pp. 845-863, 2007.

[9] L. Jaillet, J. Cortes and T. Simeon, ”Sampling-based path planning on configuration space costmaps,” *IEEE Transctions on Robotics*, vol. 26, no. 4, pp. 635-346, 2010.

[10] Y. Mao, L. Dou, J. Chen, H. Fang, H. Zhang, H. Cao, ”Combined complete coverage path planning for autonomous mobile robot in indoor environment,” in *Asian control conference*, pp. 1468-1473, 2009.

[11] H. Choset, ”Coverage for robotics-a survey of recent results,” *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1-4, pp. 113-126, 2001.

[12] H. Myung, H. Jeon, W.Y. Jeong, S.W. Bang, ”Virtual door-based coverage path planning for mobile robot,” in *Lecture note in computer science (LNCS)*, vol. 5744, pp. 197-207, 2009.

[13] V. J. Lumelsky, S. Mukhopadhyay, and K. Sun, ”Dynamic path planning in sensor-based terrain acquisition,” *IEEE Trans. on Robot. Autom.*, vol. 6, no. 4, pp. 462-472, 1990.

[14] P. Jasiobedzki, ”Detecting drivable floor regions,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Pittsburgh, PA, pp. 264-270, 1995.

[15] J. Y. Park and K. D. Lee, ”A study on the cleaning algorithm for autonomous mobile robot under the unknown environment,” in *Proc. 6th IEEE Int. Workshop Robot Human Commun.*, Sendai, Japan, pp. 70-75, 1997.

[16] Z. J. Butler, A. A. Rizzi, and R. L. Hollis, ”Contact sensor-based coverage of rectilinear environments,” in *Proc. IEEE Int. Symp. Intell. Control Intell. Syst.*, Cambridge, MA, pp. 266-271, 1999.

[17] R. Araujo, ”Prune-able fuzzy ART neural architecture for robot map learning and navigation in dynamic environments,” IEEE Transactions on Neural Network, vol. 17, no. 5, pp. 1235-1249, Sep. 2006.

[18] F. Yasutomi, D. Takaoka, M. Yamada, and K. Tsukamoto, ”Cleaning robot control,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Philadelphia, PA, pp. 1839-1841, 1988.

[19] K. Lee, W.K. Chung, ”Effective maximum likelihood grid map with conflict evaluation filter using sonar sensors”, *IEEE Transactions on Robotics*, vol. 25, no. 4 , pp. 887-901, 2009.

[20] M. G. Lourdes, A. Miranda M., Lopez P. R. and Murrieta C. R., ”Exploration and Map-Building under Uncertainty with Multiple Heterogeneous robots,” in *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 2295-2301, 2011.

[21] S. Chaoxia, W. Yanqing and Y. Jingyu, ”Online Topological Map Building and Qualitative Localization in Large-scale Environment,” *Robotics and Autonomous System*, vol. 58, pp. 488496, 2010.

[22] L. Chaomin and X.Y. Simon, ”A Bioinspired Neural Network for Real-Time Concurrent Map Building and Complete Coverage Robot Navigation in Unknown Environments,” *IEEE Transactions on Neural Networks*, Vol. 19, no. 7, 2008.

[23] Yu-Cheol Lee and Wonpil Yu, ”Practical Map Building Method for Service Robot Using EKF Localization Based on Statistical Distribution of Noise Parameters,” *Int. Symp. on Robot and Human Interac. Comm.*, pp. 478-483, Japan, 2009.

[24] A.R. Ankit, H. Yohei, E. Takanori and K. Yukinori, ”Robot Mapping Using k-means Clustering Of Laser Range Sensor Data,” in *Bulletin of Networking, Computing, Systems, and Software*, vol. 1, no. 1, pp. 9-12, 2012.

[25] G. Welch, G. Bishop, An introduction to the Kalman filter, from http://www.cs.unc.edu, UNC-Chapel Hill, TR95-041, July 2006.

[26] M. Pinto, M. A. Paulo and A. Matos, ”Localization of Mobile Robots using and Extended Kalman Filter in a LEGO NXT,” *IEEE Transactions on Education*, vol. 55, no. 1, pp. 135-144, 2012.

[27] D. Napoleon and P. G. Lakshmi, ”An Efficient K-Means Clustering Algorithm for Reducing Time Complexity using Uniform Distribution Data Points,” in *Proc. of IEEE Int. Conf. on Trendz in Info. Sc. & Computing (TISC)*, pp. 42-45, 2010.

Authors

Md Nasir Uddin Laskar received his B.Sc. degree in Computer Science and Engineering from the University of Dhaka, Bangladesh in 2008. At present, he is pursuing his M.S. degree at Artificial Intelligence Lab, dept. of Computer Engineering, Kyung Hee University, Republic of Korea. His current research interests include Mobile Robotics with special focus on Robot Motion Planning and EKF-SLAM, Human Computer Interaction (HCI), and Machine Learning.

Hoang Huu Viet received the B.S. degree in Mathematics from Vinh University, Nghean, Vietnam, in 1994, and the B.S. and M.S. degrees in Computer Science from Hanoi University of Technology, Hanoi, Vietnam, in 1998 and 2002, respectively. He is now working toward a Ph.D. degree at Artificial Intelligence Laboratory, Department of Computer Engineering, Kyung Hee University, Republic of Korea. His current research interests include Artificial Intelligence, Reinforcement Learning, and Robotics.

TaeChoong Chung received the B.S. degree in Electronic Engineering from Seoul National University, Republic of Korea, in 1980, and the M.S. and Ph.D. degrees in Computer Science from KAIST, Republic of Korea, in 1982 and 1987, respectively. Since 1988, he has been with Department of Computer Engineering, Kyung Hee University, Republic of Korea, where he is now a Professor. His research interests include Machine Learning, Meta Search, and Robotics.