Proceedings of the 11th
KACC, October 1996

Robust Position Estimation Using POMDP

Daehee KANG
Electronics Dept. Cetral R&D Institute
Daewoo Heavy Industries Ltd.
6, Manseok-dong, Dong-gu
Incheon 401-010 Korea

Abstract

In this paper, we propose a new method to estimate robot position without landmark. At first, it is studied to estimate robot state using Markov decision rule. And, a matching method is discussed for estimating current position more accurately under the estimated current state. At second, we combine or fuse the matching method with the POMDP method in order to estimate the position under a dynamically changing environment. Finally we will show that our method can estimate the position precisely and robustly of which error are not cumulated through simulation results.

1 Introduction

An accurate position estimation is very important in area such as mobile robot, especially under unknown environment. Of course, it has been studied to estimate positions of mobile robot a few decades ago. In this paper, our aim is to investigate an accurate and simple positioning method using data from range sensors. We consider an example of a two-wheel driven cart (of which top view is shown in Figure 1). The cart is

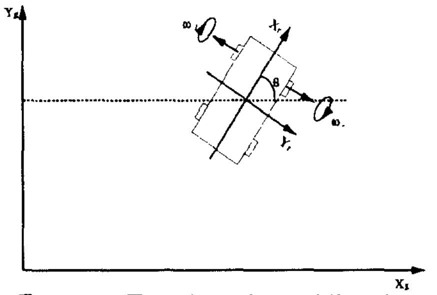

Figure 1: Top view of a mobile robot

made of three rigid bodies (the cart's platform and the two driving wheels), and it is equipped with eight range sensors and two encoders in its two wheels. It moves on a horizontal ground. Then, a parameterisation of the cart's configuration space is (x, y, θ). In this study, in order to avoid the complications that would result from introducing the mobile robot dynamic equations, we will consider only that the wheels' angular velocities ω_l and ω_r can be taken as control variables. By setting

$$U = \begin{pmatrix} v_l \\ v_r \end{pmatrix} = \begin{pmatrix} d\omega_l \\ d\omega_r \end{pmatrix} \tag{1}$$

where $v_l$andv_r are left and right wheels' velocity respectively, and d is radius of wheel. Let consider the following state vector as $X(t) = [x(t), y(t), \theta(t)]^t$ where x and y are the coordinates of the mobile robot position in the base frame, and θ is the cart's orientation angle. We then obtain the following state equation:

$$\dot{X}(t) = \begin{bmatrix} \frac{R}{2}\cos\theta & \frac{R}{2}\cos\theta \\ \frac{R}{2}\sin\theta & \frac{R}{2}\sin\theta \\ \frac{R}{2l} & \frac{-R}{2l} \end{bmatrix} \begin{bmatrix} v_l \\ v_r \end{bmatrix} = AU \tag{2}$$

Therefore, the discrete model for robot positioning is as following:

$$X^{i+1} = X^i + \tau A U^i + n \tag{3}$$

Here, n is random noise. And, assume also that the internal sensor data are corrupted by additive random noise as follows:

$$U^i = \bar{U}^i + \delta U^i = \begin{pmatrix} \bar{v}_l \\ \bar{v}_r \end{pmatrix} + \begin{pmatrix} \delta v_l \\ \delta v_r \end{pmatrix} \tag{4}$$

where $\bar{U}^i$ is the undisturbed data or the true, and δU^i is the disturbance. Now, we assume a Gaussian distribution for δv^i, that is,

$$\begin{aligned} E\left[U^i\right] &\equiv \bar{U}^i \\ V\left[U^i\right] &\equiv E\left[(U^i - \bar{U}^i)(U^i - \bar{U}^i)^t\right] \\ &= \begin{bmatrix} (\sigma^i_{v_l})^2 & 0 \\ 0 & (\sigma^i_{v_r})^2 \end{bmatrix} = Q \end{aligned} \tag{5}$$

where $E[a]$ denotes the expectation of a, and Q is the covariance matrix of U^i. Substituting Eq.(4) into Eq.(3), we calculate the mean and the covariance matrix of X^i.

$$\begin{aligned} E\left[X^i\right] &\equiv \bar{X}^i = E\left[X^{i-1}\right] + \tau A E\left[U^{i-1}\right] \\ V\left[X^i\right] &\equiv E\left[(X^i - \bar{X}^i)(X^i - \bar{X}^i)^t\right] \\ &= V\left[X^{i-1}\right] + \tau^2 A V\left[U^i\right] A^t + \mathcal{N} \end{aligned} \tag{6}$$

Here, τ is the sampling time of sensors, and $\mathcal{N}$ is a covariance of random noise n.

Consequently, the positioning error may be very large and updated always, because the covariance of the position is increased due to their cumulative effect. The major errors are generated due to followings:

- Slipping motions of wheels
- Resolution of encoder
- Mechanical parameter error
- Radius variation of wheel due to load change
- Road condition

In order to prevent the error from being updated, many researchers has usually used the landmark. We, in this paper, propose new methods to estimate robot position without landmark. At first, it is studied to estimate robot state using Markov decision process. And, a method is discussed for estimating current position more accurately under the estimated current state. Finally we will show that our method can estimate the position precisely and robustly even though the dynamically changing environment.

2 Mobile Robot Model

In case of utilizing sensor data from only encoders, it is possible to calculate the position and the orientation using "dead reckoning method" which has been widely used. However, its cumulative errors on the robot orientation and position are propagated on all three coordinates. In order to be able to state the problem, it is necessary to analyze descriptively models of the mobile robot including internal sensors. When mobile robot

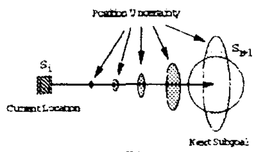

Figure 2: The positioning uncertainty

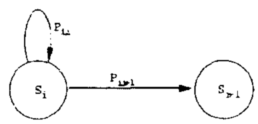

Figure 3: A model of mobile robot

moves from current subgoal(s_i) to next subgoal (s_{i+1}), the robot can be arrived at the subgoal with a probability (refer Figure 2). From now, we calculate the probabilities. When we assume that the covariance of v_l is the same as v_r's, Eq.(6) can be changed:

$$\begin{aligned} V\left[U^i\right] &= \sigma_v^2 I \\ V\left[X_{s_i}^{s_{i+1}}\right] &= V\left[X_{s_{i-1}}^{s_i}\right] + n\tau^2\sigma_v^2 AA^t + \mathcal{N} \\ n\tau &= \frac{l}{v} \end{aligned} \quad (7)$$

where $\sigma_v = \sigma_{v_l} = \sigma_{v_r}$, l is the distance between both subgoal, and v is the speed. The sensor data are corrupted by Gaussian noise, so that we can assume that X has also a Gaussian distribution with covariance matrix which is given by Eq.(7). The covariance matrix ($V\left[X_{s_i}^{s_{i+1}}\right]$) is not diagonal, but becomes semi-positive definite because $n\tau^2\sigma_v^2 AA^t$ is symmetric and semi-positive definite matrix. Its diagonalization is represented by

$$\begin{aligned} V\left[X_{s_i}^{s_{i+1}}\right] &= U^i \Lambda^i U^{i^t} \\ \Lambda^i &= diag(\lambda_1^i, \lambda_2^i, \lambda_3^i) \end{aligned} \quad (8)$$

Here, $\lambda_j^i (j = 1, \cdots, 3)$ are the eigen values and U^i is an orthogonal matrix of covariance. The scalar variance in the direction indicated by a column vector of U^i is the eigen value. At least, one of the eigen values is to be zero because the rank of covariance matrix is not full. Therefore, let's set such as follows:

$$\begin{aligned} \sigma_x^{i^2} &= \lambda_1^i, \qquad \sigma_y^{i^2} = \lambda_2^i \\ \lambda_1^i &\geq \lambda_2^i \geq \lambda_3^i = 0 \end{aligned} \quad (9)$$

Therefore, robot location is considered by a position probability density function as follows:

$$p(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} exp\left[-\frac{1}{2}\left(\frac{(x)^2}{\sigma_x^2} + \frac{(y)^2}{\sigma_y^2}\right)\right] \quad (10)$$

When a mobile robot moves to the next subgoal S_{i+1} from a current location S_i, the probability to be able to arrive at the next subgoal is calculated as followings:

$$P_{i,i+1} = \iint_{S_{i+1}} p(x, y) dx dy \quad (11)$$

where S_{i+1} is the area of next subgoal, and (x, y) is coodinates of S_{i+1}. And assuming that R is radius of S_{i+1}, Eq.(11) has a solution, $P_{i,i+1}$, which is simplified as the following:

$$\begin{aligned} P_{i,i+1} &\cong 1 - exp\left[-\frac{R^2}{2\sigma^{i^2}}\right] \\ \sigma^i &= \frac{\sigma_x^i + \sigma_y^i}{2} \end{aligned} \quad (12)$$

Consequently, when a mobile robot moves to a subgoal from current location, the robot can arrive at the goal with the probability $P_{i,i+1}$ while it can move to the other point with a probability $1 - P_{i,i+1}$ and stay at the current location with a probability $P_{i,i}$. If subgoal is only one, the robot must be stayed either at the current location with a probability $P_{i,i}$ or at the goal with a probability $P_{i,i+1}$. Therefore, we can figure out the probability process as Figure 3.

3 Robot state estimation

This section describes a state estimation method using Partially Observable Markov Decision Processes (POMDP). POMDP generalize the MDP(Markov Decision Processes) framework to the case where the agent must make its decisions in partial ignorance of its current situation. We apply the POPDM to estimate a current state of mobile robot. In here, the state of robot means the robot position or current subgoal at which mobile robot is located. As an example, Figure 4 shows a small POMDP with 2 states, 2 observations, and 1 action. Each probability of state transitions with respect to the action is that $P_{0,0} = 0.2$, $P_{0,1} = 0.8$, and $P_{1,1} = 1$. If the agent knows that it is in s_0 at time t and then takes an action and observes o_2, it can be certain that it has remained in s_0. However, if it observes o_1, it is possible that it is in either s_0 or s_1. Table 1 summarizes all 4 possible outcomes. These outcomes are mutually

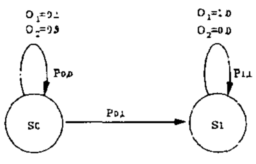

Figure 4: A POMDP

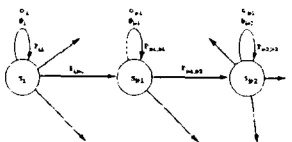
Figure 5: Path expressed by POMDP

Table 1: Probability of all outcomes

resulting state	*observation*	*probability of event*
s_0	o_1	$0.2 \times 0.1 = 0.02$
s_0	o_2	$0.2 \times 0.9 = 0.18$
s_1	o_1	$0.8 \times 1.0 = 0.80$
s_1	o_2	$0.8 \times 0.0 = 0.00$

exclusive and their total probability becomes to one. Given that the agent observed o_1, the probability that the state of robot is s_1 is as followings:

$$\begin{aligned} P(s_1|o_1) &= P(o_1|s_1)P(s_1)/P(o_1) \\ &= 0.8/(0.02+0.8) \cong 0.976 \end{aligned}$$

The probability of being in s_0 is $0.02/(0.02 \times 0.8) \cong 0.024$. Therefore, we can estimate that robot is in s_1. In section 2, we have established a probability model for mobile robot, that is, a probability of state transition between subgoals. And, a global path generated by a global path planner is usually a sequence of subgoals. By combining with them, a path for POMDP may be figured such as Figure 5 if we can obtain the observation probability in each state from environment model. And an action, θ_i, denotes a command to move from a subgoal s_i to the next subgoal s_{i+1}.

3.1 Notation for POMDP

A POMDP problem can be defined by a finite set states, $\mathcal{S}$, a finite set of action, $\mathcal{A} = \{\theta_1, \cdots, \theta_n\}$, and a finite set of observations, $\mathcal{O} = \{o_1, \cdots, o_n\}$. These sets are related with the following two functions. The transition function, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \longrightarrow \Pi(\mathcal{S})$, defines the effects of the various actions on the state of the environment. In here, $\Pi(\cdot)$ represents the set of discrete probability distributions over a given finite set. The notation $\mathcal{T}[s,a,s']$ represents the probability that state s' will result from taking action a in state s, that is, $P(s'|s,a)$. The observation function $\mathcal{O} : \mathcal{S} \times \mathcal{A} \longrightarrow \Pi(\mathcal{O})$ specifies the observation model. That is, $\mathcal{S}[s',a,o]$ is $P(o|s',a)$ which is the probability of observing o in state s' after taking action a.

3.2 Estimation for Robot State

We know $\mathcal{T}[s,a,s']$ from mobile robot model of Chapter 2, which is $P_{i,i+1}$. And also, $\mathcal{O}[s',a,o]$ has been given because we have already established a database of environment model. So that, the next state can be estimated/computed recursively as follows:

$$s_{i+1} = \underset{s'}{Max}\left(b[s']\right)$$

$$b[s'] = \left(\mathcal{O}[s',a,o]\sum_{s\in\mathcal{S}}\mathcal{T}[s,a,s']\,b[s]\right) / P(o|b,a) \quad (13)$$

The denominator makes the total probability of the resulting belief state normalized to one.

$$\begin{aligned} P(o|b,a) &= \sum_{s'\in\mathcal{S}} P(s',o|b,a,) \\ &= \sum_{s'\in\mathcal{S}}\left(\mathcal{O}[s',a,o]\sum_{s\in\mathcal{S}}\mathcal{T}[s,a,s']\,b[s]\right) \quad (14) \end{aligned}$$

In here, $b[s]$ is a belief state which is a representation of the mobile robot's current state given its past history of actions and observations. In other word, $b[s]$ represents the probability that the current state is $s \in \mathcal{S}$. By the definition of the probability distributions, $b[s] \geq 0$ for all $s \in \mathcal{S}$, and $\sum_s b[s] = 1$. For the agent to keep its belief state up to date by Eq.(13), it must start with a known belief state. That is, it must have some initial probability distribution, b_0. We set always the initial belief state $b[s_0] = 1$, where s_0 is a start state (start point) of mobile robot path.

3.3 Case study

For an example, let's consider the simple environment of Figure 6. Figure 7 shows the results of extracting subgoals by quadtree decomposition method. Assume that mobile robot has four sensor units: o_0, o_1, o_2, and o_3 are sensor units of front, left, rear, and right side on mobile robot, respectively. And, assume that the probability of sensing object's existence, when there is an object at the sensor direction, is 0.9 under considering noise, and that the path is generated such as a sequence of "$s_0 \longrightarrow s_6 \longrightarrow s_{15} \longrightarrow s_6 \longrightarrow s_{11}$" (refer to Figure 7. Table 2 shows observations at each state whenever robot moves about a discrete grid by executing an action which is a_0: **move the next subgoal**, a_1: **turn right**, a_2: **turn left**, or a_3: **stop**. The number of the Table 2 denotes probability which each sensor unit senses/detects an object. Estimate robot states using Eq.(13) and Eq.(14).

At first, we need to compute $\mathcal{T}[s,a,s']$ from mobile robot model of section 2 under assumption that the mobile robot can move two states for one action: For an example, if the mobile robot performs a_0 at s_0, it is probable to move to s_1 or s_2 or to stay in s_0.

$$\mathcal{T}[s,a,s'] = P(s'|s,a), \quad a \in \{a_0, a_1, a_2\}$$

- a_0 for $s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4$

$$\begin{aligned} P(s_{i+1}|s_i,a_0) &= P_{i,i+1} \\ &= \frac{1}{\pi}\left(1 - exp\left[-\frac{R^2}{2\sigma^2}\right]\right) \\ P(s_i|s_i,a_0) &= P(s_{i+2}|s_i,a_0) = P_{i,i} \\ &= 1 - P_{i,i+1}/2 \end{aligned}$$

- a_0 in s_5

$$\begin{aligned} P(s_6|s_5,a_0) &= \frac{1}{\pi}\left(1 - exp\left[-\frac{R^2}{2\sigma^2}\right]\right) \\ P(s_5|s_5,a_0) &= P(s_7|s_5,a_0) = P(s_{12}|s_5,a_0) \\ &= 1 - P_{i,i+1}/3 \end{aligned}$$

- And the other states have the same processes as the above one, including the follows:

$$\begin{aligned} P(s_i|s_i, a_1) &= 1 \\ P(s_i|s_i, a_2) &= 1 \end{aligned} \tag{15}$$

At second, we also need to calculate $O[s', a, o]$.

$$O[s', a, o] = P(o|s', a)$$

Events which is for each sensor unit to detect object depend on environment and its state, but reciprocally independent. In other word, the observations from sensor units are mutually independent. Therefore, $P(o_0, o_1, o_2) = P(o_0)P(o_1)P(o_2)P(o_3)$. After acting a_0 at S_5,

- If only o_2 was observed,

$$\begin{aligned} P(o|s_5, a_0) &= (1-0.1)\times(1-0.9)\times 0.9\times(1-0.9) \\ P(o|s_6, a_0) &= (1-0.1)\times(1-0.1)\times 0.9\times(1-0.1) \\ P(o|s_7, a_0) &= (1-0.1)\times(1-0.9)\times 0.9\times(1-0.9) \\ P(o|s_{12}, a_0) &= (1-0.1)\times(1-0.9)\times 0.9\times(1-0.9) \end{aligned}$$

- If o_1 and o_2 were observed,

$$\begin{aligned} P(o|s_5, a_0) &= (1-0.1)\times 0.9\times 0.9\times(1-0.9) \\ P(o|s_6, a_0) &= (1-0.1)\times 0.1\times 0.9\times(1-0.1) \\ P(o|s_7, a_0) &= (1-0.1)\times 0.9\times 0.9\times(1-0.9) \\ P(o|s_{12}, a_0) &= (1-0.1)\times 0.9\times 0.9\times(1-0.9) \end{aligned}$$

- The other cases compute such as the same as the above.

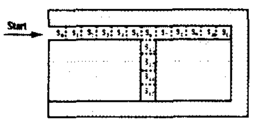

Figure 6: Environment for case study

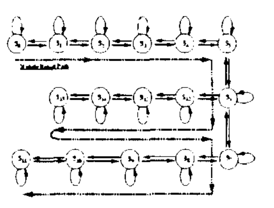

Figure 7: Robot path for case study

4 Position Estimation under Known Environment

The position of wheel-type mobile robot on the flat surface in the time step k can be presented as $P(x_k, y_k, \theta_k)$, relative to a global coordinate system. Here, θ_k is the robot's orientation. It is possible to calculate not only the position, but also the orientation using "dead reckoning method" which has been widely used. But it has several problems, such as large estimation errors and their update. In this section, our aim is to investigate of an accurate and simple positioning method using data from ultra-sonic sensors.

At first, we assume that there are some objects in known environment as (Figure 8), and that surface equations of the objects can be given as Eq.(16).

$$f_k : a_k x_g^k + b_k y_g^k = c_k \tag{16}$$

Table 2: Probability of observation and action at all states

current state	*action*	*observation*			
		o_0	o_1	o_2	o_3
s_0	a_0	0.1	0.9	0.1	0.9
s_1	a_0	0.1	0.9	0.1	0.9
s_2	a_0	0.1	0.9	0.1	0.9
s_3	a_0	0.1	0.9	0.1	0.9
s_4	a_0	0.1	0.9	0.1	0.9
s_5	a_0	0.1	0.9	0.1	0.9
s_6	a_1	0.1	0.9	0.1	0.1
s_6	a_0	0.1	0.1	0.9	0.1
s_{12}	a_0	0.1	0.9	0.1	0.9
s_{13}	a_0	0.1	0.9	0.1	0.9
s_{14}	a_0	0.5	0.9	0.1	0.9
s_{15}	a_2	0.9	0.1	0.9	0.9
s_{15}	a_2	0.1	0.9	0.9	0.9
s_{15}	a_0	0.1	0.9	0.9	0.9
s_{14}	a_0	0.1	0.9	0.9	0.1
s_{13}	a_0	0.1	0.9	0.9	0.1
s_{12}	a_0	0.1	0.9	0.9	0.1
s_6	a_1	0.9	0.1	0.1	0.1
s_6	a_0	0.1	0.9	0.1	0.1
s_7	a_0	0.1	0.9	0.1	0.9
s_8	a_0	0.1	0.9	0.1	0.9
s_9	a_0	0.1	0.9	0.1	0.9
s_{10}	a_0	0.5	0.9	0.1	0.9
s_{11}	a_3	0.9	0.9	0.1	0.9

Here, the subscript g of x_g^k represents global coordinates. Now, when distance between a object ($object_j$) and a sensor ($sense_i$) is measured as $l_{i,j}$, it is able to calculate simply the object's position on robot coordinate, such as $({}_r x_j, {}_r y_j) = (l_{i,j}\cos(\theta_i), l_{i,j}\sin(\theta_i))$. Matching the obtained object's positions with the given equations, we can estimate a robot position, that is to find out a transformation matrix of robot coordinates for the sensed object's positions to be located on the object surface. In other words, the robot position (transformation matrix of robot coordinate) can be calculated minimizing the shortest distance between the measured point P_k and the object surface as followings:

$$\begin{aligned} F(T_g^r) &= \sum_{i=1}^{n} \|P_i - \acute{P}_i\| \\ &= \sum_{i=1}^{n} \frac{|(\vec{a}_i, R_r\vec{x}_i + \vec{t}) - c_k|}{\|\vec{a}_i\|} \end{aligned} \tag{17}$$

Here,

$$T_g^r = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & t_x \\ R_{21} & R_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

where ${}_r\vec{x}_i = ({}_r x_i, {}_r y_i)^t$ is measured, and $\vec{a}_i = (a_i, b_i)^t$ is given as environment parameters. R is orthogonal matrix, and (,)is inner product operator. By the way, minimization of $F(T_g^r)$ is the same as minimizing $\sum_{i=1}^{n} \frac{((\vec{a}_i, R_r\vec{x}_i + \vec{t}) - c_k)^2}{(\vec{a}_i)^2}$, therefore, Eq.(17) becomes to be a optimization problem as following:

$$\begin{array}{lll} minimize & \Phi(\Theta) = \Theta\tilde{Q}\Theta^t - 2\Theta\hat{Q} & \\ constraints & RR^t = I, \det(R) = 1 & (18) \end{array}$$

Here,

$$\begin{aligned} \Theta &= (R_{11}, R_{12}, R_{21}, R_{22}, t_x, t_y) \\ \tilde{Q} &= \sum_{i=1}^{n} \frac{Q_i^t Q_i}{\|\vec{a}_i\|^2} \\ \hat{Q} &= \sum_{i=1}^{n} \frac{Q_i^t c_i}{\|\vec{a}_i\|^2} \\ Q_i &= (a_i x_i, a_i y_i, b_i x_i, b_i y_i, a_i, b_i) \end{aligned} \quad (19)$$

In Eq.(19), Θ can be represented as function of $\hat{p} =$

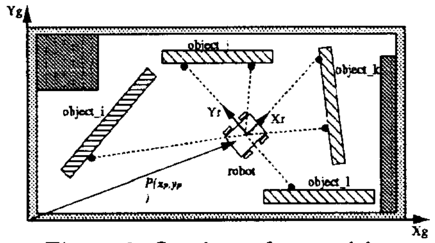

Figure 8: Sensing robot position

$(\theta, t_x, t_y)^t$ which θ is a direction angle of robot. For eliminating the constraint condition, the parameter Θ is converted with $\hat{p}$. And then, we calculate the gradient vector g_k and update the Hessian matrix of the index function $\Phi(\hat{p})$ to obtain the second-order approximation to the objective function $\Phi(\hat{p})$ and to solve the function iteratively. The iterative solution and the Hessian matrix is updated as follows:

$$\begin{aligned} \delta p &= \hat{p}_k - \hat{p}_{k-1} \\ H_{k+1} &= H_k - \frac{1}{\delta p^t H \delta p} H \delta p \delta p^t H \\ \hat{p}_{k+1} &= \hat{p}_k + H_k \delta p(trace(\Theta^t \hat{Q})) \end{aligned} \quad (20)$$

Consequently, a current position and orientation of mobile robot is obtained optimally by matching the sensed data with environment data.

4.1 Simulation Results

Mobile robot move origin point to final place through middle gates(P1,P2) - here, origin point is (0,0), final place is (85,85), and middle points are P1(15,15) and P2(85,15), where the units of numbers are 0.1 meter. At first, we tested dead-reckoning method on this environment. The results are shown at Figure 9, that is, the solid line is the actual position and the dotted line represents the estimated position by dead-reckoning method. It shows that the position error to be updated and very large and that, especially, position errors are affected by the orientation error seriously.

At second stage, we apply dead-reckoning until robot will arrive at the point P2 and then, the proposed probabilistic method is used at P2 for robot positioning because of being able to apply the method after detecting environment features such as corner points, that is, reference points. After sensing edges and corner point, Under the above situation, our least square method is

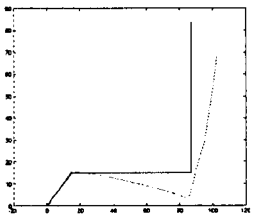

Figure 9: Positioning by dead-reckoning method

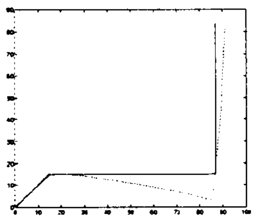

Figure 10: Least square under known environment

applied at P2. The results is better than the probabilistic approach, for example, the estimated coordinates is (85,14.03). And we apply our iterative method at P2 under same condition, the estimated coordinates is (85,14.1). As shown in Figure 10, our approach made the estimated path very accuracy even if applied only one time.

5 Position Estimation under Partially Known Environment

The proposed methods of Section 4 have some problems such as follows:

- Complex environment can have similar points too many to have only one optimal solution.
- It is necessary to know perfectly the environment.
- So that, the methods don't deal with partially known or dynamically changing environment.
- How to deal with the problem of case that ultrasonic sensor failed or missed to sense objects because the sensor can measure only objects perpendicular to the beam,
- The methods require three observations at least for having one solution.

From now, let's solve the above problems by combining the position estimation method with robot state estimation method (POMDP). In order to solve the first problem, we apply the position estimation methods under a constraint, that is, robot position is within the region of the current state which is computed by POMDP. The second and the third problems can be also solved ignoring the unknown object, that is, we deal with only objects which there can be in the current state estimated from POMDP. So that, the unknown objects are rejected from the sensed data. In fourth and fifth problem, our proposed methods have a number of solution when outer product of the measured vectors is zero (${}_r\vec{x}_i \times {}_r\vec{x}_j = 0$, refer to Eq.(17)), or when the number of measured data is only one. In the case, we determine the position as following equation:

$$\min_{t_X} \|t_X - C_X\| \quad (21)$$

where t_X, (t_x, t_y), is a vector with two elements of $\hat{p}$ in Eq.(20) and C_X is a vector from origin coordinate to a center coordinate of current state.

6 Simulation Results and Discussion

Under an environment of Figure 6, we simulate the proposed method, combining or fusing the matching method with the POMDP method. Firstly dead-reckoning method is applied between a current subgoal and the next subgoal until robot arrives at the next subgoal and estimates the current position.

In Figure 11, the dotted line represents the actual path of mobile robot, and the dotted line is the estimated position. It shows that the estimated errors are not updated.

In Figure 12, we simulate how the estimation operate under environment in where obstacle exists. Firstly, our POMDP estimates the state of robot and detects the unexpected obstacle. Therefore, the information obtained from the obstacle is rejected, and then the method for positioning applied. So that, Our method can estimate robustly the robot position without depending upon the existence of obstacle.

Figure 13 represents a weak point of our method. That is, when an obstacle is located on the same direction with the known object, the POMDP can not recognize if the obstacle is a known or unknown, so that our method looks on the obstacle as the known and occurs the estimated error bigger than dead reckoning method. However, the error is also reduced only when the obstacle disappears.

Consequently we can assert that our method estimate the position of mobile robot precisely and also robustly even if an environment is changed slightly.

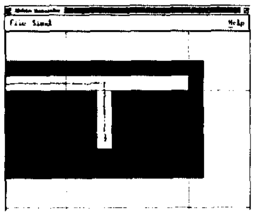

Figure 11: Estimated position of Case study

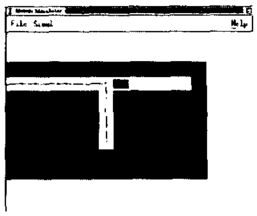

Figure 12: Obstacle 1

7 Conclusion

We have proposed, studied and simulated the robot positioning methods, which are not updated and not so large. Also, by detecting features of real world and using them instead of land-marks, we showed to be able to estimate the position of mobile robot precisely, and also showed the robustness with respect to the change of environment.

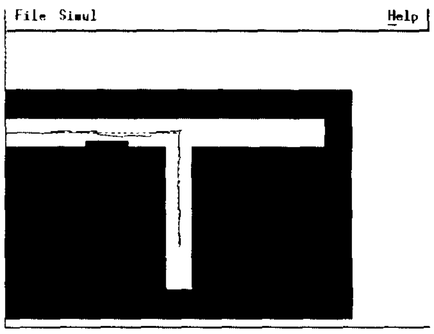

Figure 13: Obstacle 2

References

[1] Alexander Zelinsky, " A mobile robot exploration Algorithm", IEEE Transaction on R & A, Vol.8, No.6, Dec. 1992

[2] D. Kang, Ren C. Luo, H. Hashimoto, H. Harashima, " Position estimation for mobile robot under uncertain environment", MFI'94, Las Vegas, Oct. 1994

[3] A. M. Flynn, " Combining sonar and infrared sensors for mobile robot navigation", Int. J of Robotics Research, Vol.7, No.6, December,1988

[4] Ren C. Luo, " Data fusion in robotics and machine intelligence", Academic Press, 1992

[5] B.S.Y.Rao, H.F.Durrant-Whyte, J.A.Sheen, " A fully decentralized multi-sensor system for tracking and surveillance", Int. J of Robotics Research, Vol.12, No.1, pp 20-44, February,1993

[6] R.C.Smith, P.Cheeseman, " On the representation and estimation of spatial uncertainty", Int. J of Robotics Research, Vol.5, No.4, pp 56-68, 1986

[7] M.Hashimoto, F.Oba,Y.Fujikawa, " Position estimation method for wheeled mobile robot by integrating laser navigation and dead-reckoning system", JRSJ Vol.11, No.7, pp. 96-106, October, 1993

[8] Y.Suzuki, N.Kunimoto " Bayes Stochastics and Application", Univ. of Tokyo Press, 1992

[9] F. R. Noreils, R. Prajoux, " From planning to execution monitoring control for indoor mobile robots", Proc. of IEEE International Conf on Robotics and Automation, pp. 1510-1517, 1991

[10] Judea Pearl, "Probabilistic Reasoning in Intelligent Systems", Morgan Kaufmann Pub. 1988.

[11] R.C.Luo,M.G.Kay, "Multisensor Integration and Fusion in Intelligent Systems", IEEE Tran. Sys.,Man and Cybernetics, vol. Smc-19, No. 5,pp. 901-931

[12] A.M.Sabatini, "Active Hearing for External Imaging Based on an Ultrasonic Transducer Array", IROS, July 1992., pp. 829-834.