Comparison of Two Strategies of Path Planning for Underwater Robot Navigation Under Uncertainty

Teng Zhang, Shoudong Huang, Dikai Liu
Faculty of Engineering and Information Technology
University of Technology, Sydney
Sydney, Australia
{Teng.Zhang, Shoudong.Huang, Dikai.Liu}@uts.edu.au

***Abstract*—This paper considers path planning for underwater robot in navigation tasks. The main challenge is how to deal with uncertainties in the underwater environment such as motion model error and sensing error. To overcome this challenge, two high level control methods have been presented and compared, which are based on the Model Predictive Control (MPC) strategy and the Partially Observable Markov Decision Process (POMDP) model, respectively. Navigation time, collision frequency, energy consumption and accuracy in localization are used as the assessment criteria for the two methods. It is shown that the MPC-based method is more efficient for our application scenarios while the POMDP-based method can provide more robust solutions.**

***Keywords*—path planning, navigation, uncertainty, MPC, POMDP.**

I. Introduction

Underwater robot is a great application for robotics because working underwater is both dangerous and difficult for humans. Bridge piles cleaning is an important task for bridge inspection, maintenance, and rehabilitation. Fig. 1 shows the image of a bridge with piles covered by some marine growth. Carrying out high pressure underwater cleaning of piles can be a dangerous and exhaustive task for human and hence making use of underwater robot equipped with a water blasting gun can be beneficial and has large application potential. A basic requirement for such an underwater robot is that it must be able to move from its initial position to the destination (the pile that needs cleaning) without collision. Since there are a lot of uncertainties involved, such as motion error due to water current, the sensing errors, etc., the problem requires solving is path planning of underwater robot under uncertainties.

Approaches of path planning under deterministic action or full knowledge of the robot's state are quite mature and broadly classified into three categories: roadmap-based methods, cell decomposition-based methods and potential field-based methods [1]. In the past decades, the three kinds of methods have been successfully applied to the ground robot like home cleaning robots, automated wheelchair, museum-guide robot, and so on [2]. Path planning for underwater robot navigation is a more challenging task owing to the positioning uncertainty. Underwater robot suffers large motion error due to roughly known hydrodynamic coefficients. Localization using GPS is impractical and localization by other onboard sensors is not very accurate as well. It means the robot pose is only partially known. So the above three kinds of methods need to be modified to adapted to the path planning for underwater robot navigation under uncertainties.

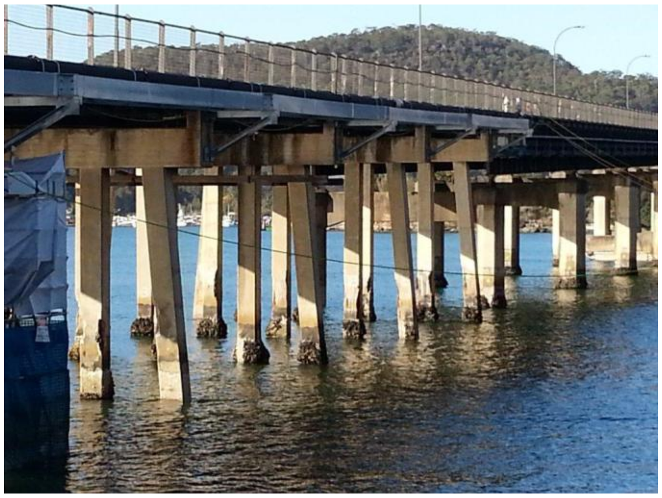
Figure 1. A bridge with piles requiring cleaning

In order to alleviate the difficulties in the navigation under uncertainties, many approaches have been proposed and some of them have been proved to be successful and effective in simulation and practice such as [2], [3], [4].

Burns and Brock [5] propose a sampling-based planner that incorporates sensor uncertainty into the planning process, determines an optimal path and then give a control policy. Van Den Berg and Overmars [6] present similar approaches for path planning by using different evaluation function for paths, which also considers motion uncertainty. Roijers *et al.* [7] propose a motion planner based on multi-objective Markov Decision Process, which considers the motion uncertainty but ignores the observation uncertainty. Recently, Vadim *et al.* [8] propose a planning approach including the estimation of the position of landmark. These approaches are based on the Discrete Stochastic Process, which regards robot's motion as sequential actions that are not deterministic. Furthermore, the three kinds of basic methods for path planning without uncertainty are also subtly exploited as well.

In fact these planners mentioned above try to trade off between the size of the model (state space, action space, observation space) and the number of the steps that the robot looks ahead because the large-size model and the large number of steps usually require tremendous computing time. Discretization and the finite steps look-ahead policy are the popular techniques for making the problem tractable. MPC, as a representation of finite steps look-ahead policies, associated with extended Kalman Filter can handle the problem of planning with the continuous working space over finite steps [9].

In theory, POMDP is a general framework for planning under uncertainty over infinite steps and it may be useful for many application areas such as industry applications and business applications [10]. POMDP was regarded as impractical a decade ago due to its high computational complexity. However, the point-based POMDP algorithms have made dramatic progress in recent decade [2], [11], [12]. POMDP provides the optimal solution over the infinite steps but it is usually used for the discrete model. MPC and POMDP simplify path planning under uncertainty from different perspectives. In this paper, we make some comparison between the two strategies in our application scenarios and try to identify which is more suitable for our problem.

The remainder of this paper is organized as follows. The details of the navigation problem considered in this paper are presented in Section II. The MPC-based method is presented in Section III. The POMDP-based method is described in Section IV. Results and comparison based on simulation are shown in Section V. Section VI addresses conclusion and future work.

II. THE DETAILS OF NAVIGATION PROBLEM

We consider the scenario in 2D, ignoring the vertical direction. This simplification is acceptable for three reasons. The first reason is that the shape of obstacles in our scenario is almost the same at different height (cuboid). The second reason is that the underwater robot can adjust itself to different height. The third reason is that it is relatively easy to obtain the vertical position of an underwater robot. In this paper, kinematic and dynamic constraints has not been considered in this paper.

The 2D environment is illustrated in Fig. 2, where the squares and strips are bridge piles. The red square is the pile requires cleaning which is the destination of the robot. The blue point is the actual position of the robot. The exact initial pose of the underwater robot is unknown but its probability distribution is given. We consider configuration space which regards the robot as a dot and enlarges the size of obstacles.

A. Objectives and evaluation function

The main objective is to drive the underwater robot quickly arriving at the neighborhood of the specific pile without collision. The second objective is to minimize the uncertainty of the underwater robot's pose in the navigation. The third objective is to minimize energy consumption which is assumed to be proportional to the total degree of the underwater robot's steering angle. The task can be summarized as four objectives: (1) arriving at the neighborhood of the specific pile in short time, (2) avoiding obstacles, (3) localizing itself precisely, and (4) reducing the energy consumption.

For these purposes, an evaluation function for different methods is proposed as following:

$$Eva(M) = (E(t), E(e), E(q), E(c))^T. \quad (1)$$

In this function, M represents method, t represents the total time in navigation, e is an index that quantifies the uncertainty of the robot's pose, $c = 1$ indicates the collision happens while $c = 0$ indicates the robot avoids obstacles in the navigation and q symbolizes the total degree of steering angle. Furthermore, E is the expectation operator. A method has better performance if its value in the evaluation function is smaller.

B. Assumptions

The map as shown in Fig. 2 is assumed to be given in advance. The squares and strips are obstacles. The squares are regarded as landmarks as well. The red one is the destination which is given. The probability distribution of the underwater robot's pose is given. The underwater robot can obtain an observation that indicates the approximate distances between its current position and the square piles within the sensor range.

The motion model of the robot is given which contains uncertainty. For simplification, we only consider the case that the velocity of the underwater robot is fixed. Like other methods mentioned above, navigation is regarded as a stochastic process and we assume the robot would perform one action and then get one observation in each step. Performing an action means the robot changes the heading direction in the interval $[-a, a]$ $(a > 0)$.

C. Strategies for comparison

A strategy can be defined as a mapping from information history to action. Information history indicates the pose probability distribution history, observation history and action history. Section III and Section IV introduce two strategies based on their specific model, respectively.

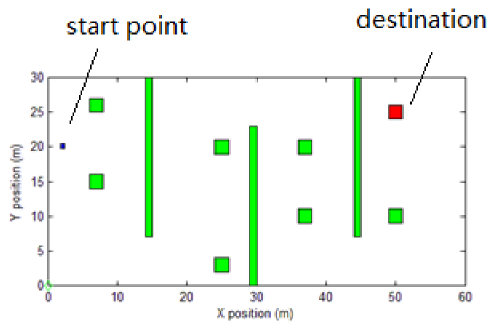

Figure 2. A map of our application scenario

III. THE MPC-BASED METHOD

The method based on MPC is inspired by the previous work [9]. The key of MPC is that "using available model to predict a few steps, find the optimal actions and execute only one step and then update the model". Under the assumption that the initial robot pose, motion error and sensing error follow Gaussian distribution, the localization and prediction can be performed efficiently.

A. Transition model and observation model

Suppose the robot's pose at the time step i is

$$X_i = (x_i, y_i, \theta_i)^T, \quad (2)$$

where (x_i, y_i) represents the position of the robot, θ_i is the heading direction. The transition model is given by

$$X_{i+1} = f(X_i, u_{i+1}) + e_i, \quad (3)$$

where f is a function depending on the kinematic model, u_{i+1} as input action is the steering angle between the time step i and the time step $i+1$, and e_i as motion error follows the Gaussian distribution $\mathcal{N}(0, R_i)$. R_i depends on the water current at the time step *i*. If the navigation time is short, this matrix is fixed.

In our proposed scenario, we assume

$$f(X,u) = \begin{pmatrix} x + v_0 cos(\theta+u) \\ y + v_0 sin(\theta+u) \\ \theta + u \end{pmatrix}, \quad (4)$$

where v_0 as constant is the displacement in the unit time.

The underwater robot is assumed to be equipped with a sensor that can obtain each distance between the robot and the piles within the sensor range. If the robot senses the j th pile, the measurement is

$$z^j = h_j(X) + d^j = ||x - L_j^x, y - L_j^y|| + d^j, \quad (5)$$

where z^j is the measurement, (x, y) is the robot's position, (L_j^x, L_j^y) is the position of the j th pile, and d^j as sensing error follows the Gaussian distribution $\mathcal{N}(0, Q_j)$ in which Q_j depends on the level of the sensor noises.

Furthermore, $Z_i = (z_i^{i_1}, z_i^{i_2}, \dots, z_i^{i_l})^T$ is called the observation at the time step i, where z_i^j indicates the measurement of the j th pile at the time step i.

B. Extended Kalman Filter (EKF)

EKF makes the MPC-based method easy to compute. If the robot's pose at the time $i-1$ follows the Gaussian distribution $\mathcal{N}(X_{i-1}, C_{i-1})$, it performs the action u_i between the time step $i-1$ and the time step i and then obtains the observation $Z_i = (z_i^{i_1}, z_i^{i_2}, \dots, z_i^{i_l})^T$ at the time step i where z_i^k indicates the measured distance between the robot and the k th pile at the time step i, the robot's pose at the time step i will follows the Gaussian distribution $\mathcal{N}(X_i, C_i)$.

The (X_i, C_i) can be computed as following:

Prediction step:

$$\bar{X} = f(X_{i-1}, u_i)$$

$$\bar{C} = \left(\nabla_X f(X_{i-1}, u_i)\right) C_{i-1} \left(\nabla_X f(X_{i-1}, u_i)\right)^T + R_i.$$

Update step:

$$K = \bar{C}\left(\nabla_X h(\bar{X})\right)^T \left(\left(\nabla_X h(\bar{X})\right)\bar{C}(\nabla_X h(\bar{X}))^T + \tilde{Q}\right)^{-1}$$

$$X_i = \bar{X} + K(Z_i - h(\bar{X}))$$

$$C_i = (I - K(\nabla_X h(\bar{X})))\bar{C},$$

where

$$h(\bar{X}) = \left(h_{i_1}(\bar{X}), h_{i_2}(\bar{X}), \dots, h_{i_l}(\bar{X})\right)^T$$

$$\tilde{Q} = \mathrm{diag}(Q_{i_1}, Q_{i_2}, \dots, Q_{i_l}).$$

The above EKF equations can be summarized as a brief expression as

$$[X_i, C_i] = g(X_{i-1}, C_{i-1}, u_i, Z_i). \quad (6)$$

For more details of the EKF formula, please refer to [9] and [13].

C. Nonlinear MPC control

In order to predict the performance in the next step without actually performing the control action and obtaining the observation, some assumptions are necessary. Similar to [10], for prediction, we assume that no new piles will be detected in the next step and the robot will acquire the observation that is most likely given the Gaussian distribution.

That is, if the robot can sense the $i_1, i_2, \dots, i_l$ th piles at the time step i, it will sense the same piles in the next step. And if the observation is Z_i at the time step i, the robot performs the action u between the time step i and the time step $i+1$ and the pose follows Gaussian distribution $\mathcal{N}(X_i, C_i)$ at the time step i, the observation at the time step $i+1$ will be

$$\bar{Z}_{i+1} = \left(h_{i_1}(f(X_i,u)), h_{i_2}(f(X_i,u)), \dots, h_{i_l}(f(X_i,u))\right)^T. \quad (7)$$

Combining (6) and (5), the Gaussian distribution of the robot's pose at the time *i+1* will be easy to predict as:

$$[\bar{X}_{i+1}, \bar{C}_{i+1}] = g(X_i, C_i, u, \bar{Z}_{i+1}). \quad (8)$$

In order to minimize the multiple objective functions in (1), we design a weighted sum of them, which is

$$J(u) := w_1 J_d(\bar{X}_{i+1}, \bar{C}_{i+1}) + w_2 J_e(\bar{C}_{i+1}) + w_3 ||u|| \quad (9)$$

$$\text{Subject to: } J_c(\bar{X}_{i+1}, \bar{C}_{i+1}) = 0, u \in [-a, a]$$

Let us explain all the terms of (8). The term $J_d(X, C)$ is the average length of the shortest path from the point in the ellipsoid

$$(x,y)\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}(x,y)^T = 4 \quad (10)$$

to the destination, where C_{ij} indicates the entry in the i th row and j th column of C. This ellipsoid covers 95% confidence region area of the actual position of the underwater robot. Furthermore, $J_d(X, C)$ can be easily computed by using sample-based method. The principle is that sample many points in the ellipsoid (9) according to the Gaussian distribution $\mathcal{N}(\bar{X}_{i+1}, \bar{C}_{i+1})$, and then compute the average length of shortest paths from the sampled points to destination. The length of shortest path can be considered as a function $Le(x, y)$, which can be computed offline by using Dijkstra's algorithm. In this part, computational complexity is polynomial time, which is practical and acceptable. Using $J_d(X, C)$ is helpful for the robot arriving to destination without falling in to the local stable point unlike some cases in the force field method.

The term $J_e(C)$ is used to quantify the uncertainty of the robot pose because

$$J_e(C) := \det\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} \quad (11)$$

is proportional to the area of the ellipsoid (9).

The term $||u||$ has two effects. One is that we want to minimize the energy consumption which is assumed to be positively correlated with the steering angle u. Another is that the transition model (3) is an approximate when the steering angle u is small.

The constraint $J_c(\bar{X}_{i+1}, \bar{C}_{i+1}) = 0$ is to guarantee collision-avoidance as far as possible. $J_c(X, C)$ is the common area of the ellipsoid (9) and the obstacles. In the proposed 2D scenario, checking if the common area of the ellipsoid (8) and the obstacles (squares and strips) is zero or not is very quick.

The second constraint $u \in [-a, a]$corresponds to the limitation of the steering angle. A practical way for the optimization problem (8) is to discretize the set $[-a, a]$. This simplification is reasonable and also reduces much computing time.

w_i $(i = 1,2,3)$ in (8) is the weight coefficient. However, it is very difficult to quantify the weight coefficient without prior knowledge. If the weight coefficients are fixed, the robot may stay at some region that locally minimize (8). Considering the main objective is that the underwater robot arrives at the destination as soon as possible, we devise a self-adaptive strategy to address this problem. The principle is simple and natural:

If $J_d(X_i, C_i) > J_d(X_{i-k_0}, C_{i-k_0})$, w_1 will be enlarged temporarily until $J_d(X_i, C_i) < J_d(\bar{X}_{i-M}, \bar{C}_{i-M})$, where i means the current time and $i - k_0$ means the last k_0 step. In our simulation in Section V, k_0 is set as 5.

The solution u_* of the optimization problem (8) is the optimal action between the time step i and the time step $i + 1$.

The MPC-based method described above only considers one step look ahead. The multi-step look ahead MPC strategy is similar, but requires more computational cost [9].

IV. THE POMDP-BASED METHOD

The POMDP-based method is a strategy for planning under uncertainty with infinite time horizon. Different from the MPC-based method using EKF, POMDP has no Gaussian distribution assumption on the initial pose, the noise on transition model and the observation model. Strictly, a POMDP model is a tuple (S, A, O, T, Z, R, r). The details are given below.

A. *Transition model and observation model*

S includes all possible robot's poses and an absorbed state *end*. The state *end* is used to end the navigation. A is the action set. T is the transition model according to (2).

$T(s, u, s') \coloneqq \Pr(s'|s, u)$ is the probability that the robot's pose would be s' under the action u if the current pose is s. The absorbed state end is to stop the whole process. Generally speaking,

$$T(destination, u, end) = 1, \forall\, u \in A \tag{12}$$
$$T(end, u, end) = 1, \forall\, u \in A. \tag{13}$$

O includes all the possible observations. Z is the observation model, $Z(o|s)$ is the probability that the robot is at the pose s and acquires the observation o. In the proposed problem, we formulate o as the distance between the robot and the nearest pile.

For the proposed scenario, S , A and O are discrete sets. Naturally, the transition model is given according to the discretization of (2):

$$T((x', y', \theta + u)|(x, y, \theta), u) \propto \exp[-(d_x^2 + d_y^2)], \tag{14}$$

where

$$d_x = x' - x - v_0 \cos(u + \theta), \tag{15}$$

$$d_y = y' - y - v_0 \sin(u + \theta). \tag{16}$$

The observation model is given according to the discretization of (4):

$$Z\big((o_1, o_2, \dots o_L)\big|(x, y, \theta)\big) \propto \prod_{i=1}^{L} \exp\left[\frac{-1}{2}\left(\sqrt{q_{i,x}^2 + q_{i,y}^2} - o_i\right)^2\right], \tag{17}$$

where L is the number of piles, o_i is the measurement that indicates the distance between the robot and the i th pile, and $q_{i,x} = x - L_x^i$, $q_{i,y} = y - L_y^i$, where (L_x^i, L_y^i) is the position of the the i th pile.

The observation model in POMDP is a little different from that in MPC. In POMDP, the event that the robot cannot sense any pile is also regarded as an observation while the method based on MPC ignores this.

B. *Reward function and discount coefficient*

Reward function $R(X, u)$ means the robot will obtain a value as reward under the action u when it is in the state x. Discount coefficient r is a constant in $(0,1)$. The POMDP solution is to maximize the total rewards.

In order to minimize the multiple objective functions in (1), reward function $R(s, u)$ can be devised as:

$$R(end, u) = 0, \tag{18}$$

$$R\big((x, y, \theta), u\big) := -R_d(x, y) - ||u|| - R_o((x, y, \theta), u), \tag{19}$$

where $R_d(x, y)$ is the length of shortest path from (x, y) to the destination. R_o is the penalty function for obstacles: if the underwater robot at the position (x, y) moves to obstacles under action u with high probability, the value of $R_o(X, u)$ will be very large. Furthermore, $R(destination, u)$ is set as a large value, which encourages that the underwater robot arrives at the destination as soon as possible.

C. *POMDP* solution

POMDP solution is a mapping from the space of the probability distribution of the robot to the action set A. The solution π can maximize the function value

$$V_\pi(b) = E\left[\sum_{i=0}^{\infty} r^i R(X_i, u_i|b, \pi)\right] \tag{20}$$

for arbitrary probabilistic distribution b. Computing the solution is done offline and it usually takes much time. This solution provides an action $u = \pi(b')$ at each step where b' is the probability distribution of the robot pose in the current step. See more details in [12] and [13].

V. RESULTS AND COMPARISON

We apply the MPC-based method and the POMDP-based method in two simulation examples. In the simulation examples, the terminal conditions are that the underwater robot arrives at the destination. The boundaries are also considered as obstacles. Furthermore, the sensor of the robot is assumed to have the omnidirectional field and the robot velocity is set as 1m/s. The length function $Le(x, y)$ of the shortest path in the MPC-based method is computed by using Dijkstra's algorithm. The sensor range in the first example (Fig.3 and Fig. 4) is assumed to be 10m while it is set as 5m in the second example (Fig. 5 and Fig. 6).

The parameters of the MPC-based method are set as the following:

$$a = \pi/4, \quad w_1 = 2, w_2 = 1, w_3 = 1.$$

$$C_0 = R_i = diag(0.001, 0.001, 0.1) \ (i = 1,2,3 \dots)$$

In the first example,

$$X_0 = (2, 20, 0)^T, \quad Q_j = 0.5 \ (j = 1,2, \dots, 8),$$

In the second example,

$$X_0 = (-3, -3, -\pi/4)^T, \quad Q_j = 0.5 \ (j = 1,2, \dots, 6).$$

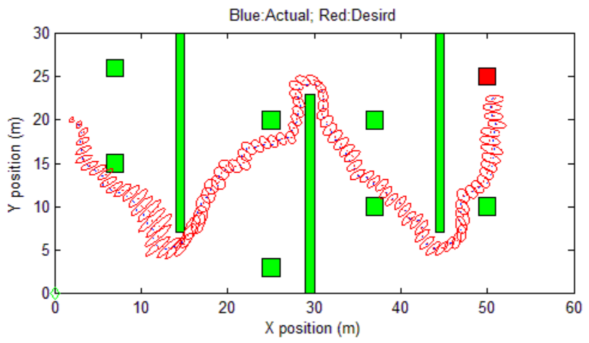

Figure 3. A simulation run by MPC

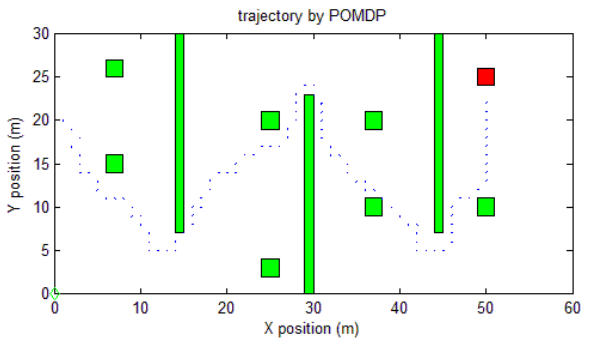

Figure 4. A simulation run by POMDP

The details of the POMDP transition model and observation model are trivial and thus omitted. In both of the first example and the second example, the underwater robot would get +10000 reward value when it arrives at the destination. Furthermore, the discount coefficient r is set as 0.98.

In the first map, the size of POMDP is:

$$|S| = 14001,$$

$$|A| = 7, |O| = 25.$$

The size of the POMDP model in the second map is much smaller:

$$|S| = 5001,$$

$$|A| = 7, |O| = 36.$$

For each example, to estimate the function value of (1), we perform 100 simulation runs under the MPC-based method and the POMDP-based method, respectively and compute the expectations.

Fig. 3 and Fig. 4 show that the trajectories by the two methods are very similar. TABLE I also illustrates this point. In the first example, $E(t)$ and $E(c)$ values from TABLE I show that the paths have small difference in length and the collision probability is also at the same level. The smaller $E(e)$ in the MPC-method means more accurate localization. The smaller $E(q)$ in the MPC-method means the smoother trajectory and the smaller power consumption.

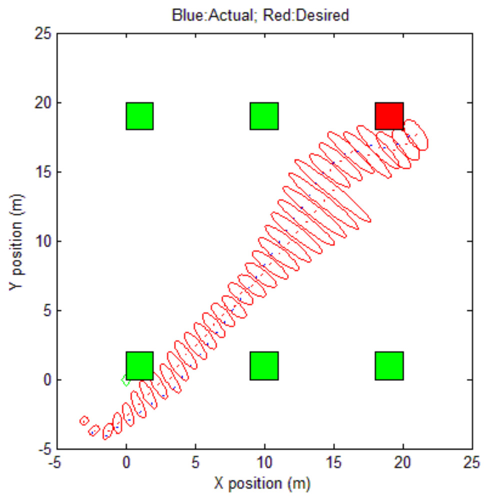

Figure 5. A simulation run by MPC in the second map

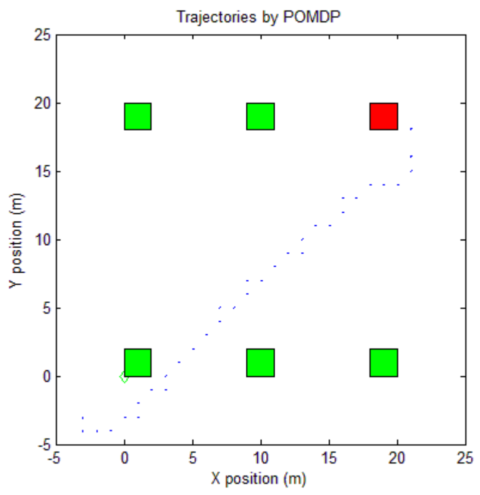

Figure 6. A simulation by POMDP in the second map

The range of view field in the second map is only 5m. The results are given in Fig. 5 and Fig. 6. The comparison of the two methods is given in TABLE II. It can be seen that the result in TABLE II, $E(e)$ and $E(q)$ by the POMDP-based method are smaller. This is interesting. One reason is that POMDP can utilize more “information” to update the robot’s knowledge while the MPC-method usually ignores the event that some piles cannot be sensed.

Furthermore, the POMDP-based method needs much more offline time to get a relative accurate solution while the MPC-based method requires less offline time. However, the POMDP-based method takes less time on the online part. Three reasons may explain the big difference on computation time between the MPC-based method and the POMDP-based method. The first reason is that the online part of the POMDP-based method is usually very simple (strictly, the complexity of the online computation depends on the number of α-vectors). The second reason is that the POMDP-based method considers the infinite steps while the MPC-based method only considers finite steps. The last reason is that the MPC-based method simplifies the process of updating the probability distribution of the robot's pose by utilizing EKF that calculates an approximation to the true probability distribution [14].

TABLE I. COMPARISON IN THE FIRST MAP

Results	**Methods**	
	MPC method	***POMDP***
E(t)	98	95
E(e)	2	2.7
E(q)	3.9	5.2
E(c)	0.02	0.02
Computing time	0.1s (online) +10m (offline)	0.01s (online) +2h (off line)

TABLE II. COMPARISON IN THE SECOND MAP

Results	**Methods**	
	MPC method	***POMDP***
E(t)	33	32
E(e)	3.1	2.7
E(q)	2.74	2.4
E(c)	0	0
Computing time	0.1s (online) +5m (offline)	0.001s (online) +2h (off line)

The MPC-based method is implemented in Matlab 2009b, and the code used to solve the POMDP problem in this paper is Sarsop, specified details of which are given in [12]. The two planners are run in a PC with 4.0GHz Intel processor and 16GB RAM.

VI. CONCLUSION AND FUTURE WORK

Two path planners for the underwater robot navigation under uncertainty have been presented and compared. The particular objectives include quick arrival, accurate localization, collision-avoidance and reducing the power consumption.

We can make a preliminary conclusion from the simulation results that the MPC-based method is very efficient and effective but its performance heavily depends on the range of view filed. The POMDP-based method can exploit more information in theory. Especially, it is more suitable for the case that the number of possible observations is few. For real application, whether to use the MPC-based method or the POMDP-based method depends on the balance between performance and computing time.

Significant effort is still required before implementing the method into the real world. Gaussian distribution in the MPC-based method may be extended to the general probability distribution. On the other hand, POMDP and MPC are high level control method, so developing a smart algorithm combing them and low level control is also necessary. Future work also includes trying to combine the local update policy from the recent POMDP algorithm [14] with the MPC-based method.

REFERENCES

[1] LaValle, Steven M. Planning algorithms. Cambridge university press, 2006.

[2] R. Simmons and S. Koenig, "Probabilistic robot navigation in partially observable environments," in *IJCAI*, 1995, pp. 1080-1087.

[3] R. Kaplow, A. Atrash, and J. Pineau, "Variable resolution decomposition for robotic navigation under a POMDP framework," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 369-376.

[4] L. J. Guibas, D. Hsu, H. Kurniawati, and E. Rehman, "Bounded uncertainty roadmaps for path planning," in *Algorithmic Foundation of Robotics VIII*, ed: Springer, 2009, pp. 199-215.

[5] B. Burns and O. Brock, "Sampling-based motion planning with sensing uncertainty," in *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 3313-3318.

[6] J. P. Van Den Berg and M. H. Overmars, "Roadmap-based motion planning in dynamic environments," *Robotics, IEEE Transactions on*, vol. 21, pp. 885-897, 2005.

[7] D. M. Roijers, J. Scharpff, M. Spaan, F. A. Oliehoek, M. de Weerdt, and S. Whiteson, "Bounded approximations for linear multi-objective planning under uncertainty," *International Conference on Physics and Astronomical Sciences, 2014.*

[8] V. Indelman, L. Carlone, and F. Dellaert, "Planning Under Uncertainty in the Continuous Domain: a Generalized Belief Space Approach." unpublished.

[9] C. Leung, S. Huang, N. Kwok, and G. Dissanayake, "Planning under uncertainty using model predictive control for information gathering," *Robotics and Autonomous Systems*, vol. 54, pp. 898-910, 2006.

[10] A. R. Cassandra, "A survey of POMDP applications," in *Working Notes of AAAI 1998 Fall Symposium on Planning with Partially Observable Markov Decision Processes*, 1998, pp. 17-24.

[11] G. Theocharous, K. Murphy, and L. P. Kaelbling, "Representing hierarchical POMDPs as DBNs for multi-scale robot localization," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, 2004, pp. 1045-1051.

[12] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces," in *Robotics: Science and Systems*, 2008, pp. 65-72.

[13] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*: MIT press, 2005.

[14] H. Kurniawati and V. Yadav, "An Online POMDP Solver for Uncertainty Planning in Dynamic Environment," in *International Symposium on Robotics Research*, 2014.