Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea - May 21-26, 2001

Towards a Meta Motion Planner A: Model and Famework

Amit Adam
Dept. of Mathematics
amita@tx.technion.ac.il

Ehud Rivlin
Dept. of Computer Science
ehudr@cs.technion.ac.il

Ilan Shimshoni
Dept. of Industrial Engineering
ilans@ie.technion.ac.il

Technion – Israel Institute of Technology
Haifa 32000 – Israel

Abstract

We address the problem of rating or comparing navigation algorithms, or more generally navigation packages. For a given environment a navigation package consists of a motion planner and a sensor to be used during navigation. The ability to rate or measure a navigation package is important in order to address issues like sensor customization for an environment and choice of a motion planner in an environment.

We develop a framework under which we can rate a given navigation package. Based on the navigation package, a partially observable Markov decision process (POMDP) is defined. Next an optimal policy to be used in this POMDP is searched for. The performance achieved under the resulting policy serves to measure the navigation package.

This paper presents the motivations for solving the problem, the model we use and the framework which we have developed. An accompanying paper [1] presents the algorithm which we use and some results.

1 Introduction

When given a navigation task in an environment, a robot will usually employ certain sensors and a certain motion planner. In this and a companion paper [1] we discuss the problem of choosing the best combination of motion planner and sensor. This paper focuses on the motivations for solving this problem, the model which we use for the problem, and the framework under which this problem may be approached. In the companion paper [1] we present an algorithm to be used for solving the problem and demonstrate results of applying this framework and algorithm to some sample problems.

We will start with an example which will clarify our motivation and the terms we will use.

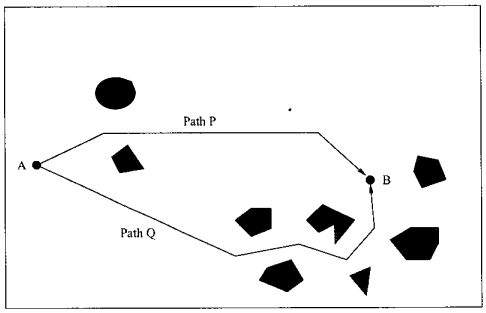

Figure 1: An example of an environment

1.1 A Motivating Example

Consider the environment shown in Fig. 1. Suppose that our robot has to move from point A to point B, without colliding with the obstacles. In order to compute its course, the robot employs a motion planning algorithm. A large number of motion planning algorithms may be employed [11], and in general each will compute a different *nominal path* from A to B. Paths P and Q in the figure are examples of two nominal paths which were generated by two different motion planners.

Should our robot use path P or should it choose path Q ? In principal, both paths are good and should be suitable for navigation. We might want to consider the shorter path between the two. However, other factors also come into play. The first is uncertainty in positioning. For various reasons (see for example [3]), we cannot expect the robot to be able to follow accurately a nominal path which was planned by the motion planner. As the robot attempts to execute a path, uncertainty in its position will develop. This uncertainty will result for example in a certain chance of hitting an obstacle. In our example, it is likely that for path P this chance is lower than for path Q. Thus, in addition to path length, other spatial characteristics such as distance from ob-

0-7803-6475-9/01/$10.00

stacles, number of turns, where in the environment the path actually traverses (e.g in the bottom region, in the center ...) etc. may also affect our choice of nominal path.

In order to reduce the uncertainty in position, the robot invokes from time to time a sensor. Many types of sensors are available [3]. Let us assume specifically that our robot is equipped with a camera and that it updates its position by comparing the current image it obtains from the camera, with a database of images of the environment that is stored in its memory. Let us assume that this database of images is given. A natural question that we now ask is the following: along which path, P or Q, will the robot be better able to compare the images it will acquire with the specific images stored in its database ? For example, it may be possible that the images in the database were all photographed from points in the bottom half of the environment. In this case, we expect the vision sensor to be more useful along path Q rather than along path P.

Once we have chosen the nominal path to follow, we are faced with the question of actually executing this path. The robot has to decide for example when it wants to invoke its sensor. It would not be efficient to invoke the sensor too often. On the other hand, we cannot let uncertainty in position grow too much (because of collisions with obstacles for example). The amount of uncertainty we are willing to allow may also vary along each different nominal path (because of variable distance to obstacles for example). Thus for each nominal path we expect to find a different sensing strategy which should be used along the path.

2 Navigation Packages

The example above shows us that a number of factors or elements interact with each other in the navigation problem. The factors we have identified are the environment itself, nominal paths in the environment, uncertainty in the position of the robot and a sensor which is used to reduce this uncertainty.

It is convenient to represent the sensor abstractly by a performance map defined on the environment. For every point p in the environment, this map states the expected performance or accuracy of the sensor if invoked at the point p. For example, in our image database sensor, this accuracy will be related to the ability to compare the image seen from the point p to one of the images in the database. By representing the sensor as a performance map we free ourselves from referring to the actual implementation of the sensor.

We use the term "navigation package" to denote a combination of all the factors mentioned above. Because the environment and robot are usually given and fixed, we will sometimes think of the navigation package as consisting of a motion planning algorithm producing nominal paths, and a sensor, which is represented abstractly by its performance map.

3 Rating Navigation Packages

In this work we discuss the problem of rating or comparing different navigation packages. The interaction between the nominal path, the performance of the sensor across the environment, the spatial features of the environment, and the uncertainty which develops in the position of the robot, result in certain measurable consequences. These may be the cost of sensing, the chances of bumping into obstacles, the chances of reaching the goal, the lengths of the paths taken etc. It is therefore natural to try to *rate* or *measure* the quality of different navigation packages for a given environment.

By being able to compare navigation packages, many interesting problems may be tackled:

- What is the best motion planner to be used with a given environment and sensing capability ? In our example, this was the question of choosing between path P and path Q. Note that not only the environment but also the sensing capability are important in making the decision.

- Suppose the sensor can be customized for a given environment. What is the preferred customization ? In our example, "customizing" may mean the choice of images to be stored in the database. How should we photograph the environment in order for the image database to constitute a useful sensor ? The answer to this question may depend on the nominal paths we plan to use.

- Given a specific sensing capability in an environment, should we use different motion planners in different regions of the environment ? For each region of the environment, the motion planner which is best combined with that region's sensing capability and spatial characteristics, should be chosen.

Notice that in each of these examples, the choice we actually have to make is between different navigation packages. In the first example the sensor is given and we change the nominal paths, thus yielding different navigation packages. In the second example different "customizations" yield different sensor performance maps, thus defining different navigation packages. Therefore it is important to be able to compare different navigation packages.

We note that the problems we have discussed are especially relevant when working with vision sensors. One reason is that the performance of vision-based sensors is highly variable and dependent on location. For example, occlusions that interfere may degrade the performance in a very well defined region. The second reason is that the issue of customization is also very relevant. For example, we may choose what images to store in the database, or we may choose where to place visual landmarks.

4 Difficulty of Rating a Navigation Package

Assume that we are given a navigation package for a certain environment consisting of a motion planner which plans nominal paths, and a sensor which would be used to update the position of the robot along the path. The first thing we should understand is how to augment the nominal path with a sensing strategy. This strategy will be employed upon actual traversal of the path. A sensing strategy is required since sensing may be a costly operation which we would like to avoid as much as possible. Some considerations that should be addressed by this strategy are:

- The performance of the sensor at the place where sensing is invoked - there is no point in invoking the sensor where it is useless.
- The required accuracy in position. In some places an accurate update may be required (for example near obstacles) while in other places accuracy may be compromised.
- The cost of sensing - how often should the sensor be invoked.

Note that the robot is uncertain of its position. As the current position is unknown, the robot cannot use the map which describes sensor performance as function of position in a straight forward way.

All of these considerations make it clear that finding a reasonable sensing strategy is not a trivial task. However, it would only be fair to rate a navigation package based on its performance under the *best* sensing strategy available for this package. Therefore, we should first find the strategy under which the robot should act when using the navigation package. The navigation package will then be rated based on the performance achieved by using it in conjunction with the strategy found.

5 Our Framework and Related Work

Before we describe our framework, let us first describe related work by other authors. Many authors have noted that the performance of a sensor varies across the environment. In [12] regions of perfect sensing have been called landmarks or "islands of truth". A map describing the sensor performance has been termed Sensory Uncertainty Field (SUF) in [18]. Quite a few works have shown the benefits of using such a map. Use of the map in motion planning has been demonstrated in [18, 7, 10]. In [15] a related notion is the information content of the environment at each configuration. Another similar idea motivated by visual servoing is described in [16]. In [4, 5] sensing management issues are discussed and the sensor performance map is a factor taken into account. Finally, in a previous work [2] we have addressed the issue of computation of such a map for vision-based sensing.

The motion planning approach taken in previous works (for example [18, 15]) is to search for a path in free space by minimizing a function which takes into account both the length of the path and the sensory uncertainty along the path. This approach involves some arbitrary decision on how to trade off between sensory uncertainty and path length. These two different factors are usually combined into one objective function by introducing an arbitrary scale factor between the two.

Both our problem and our method of solution are different. As explained above, we would like to find a method for comparing two different navigation packages. We have chosen the partially-observable Markov decision process (POMDP) framework in order to address this problem. In our framework, every navigation package defines a POMDP which expresses the sensing capability, the underlying nominal path, the developing uncertainty in position and the environment. An optimal policy under which to act in this POMDP is searched for. The expected performance under this policy is the rating of the navigation package. We are then able to compare different navigation packages by comparing the performance in the POMDPs they have defined, under the policies that were found.

As a by-product of the process of rating a navigation package, our method yields a policy under which the robot should act while navigating in the environment. This policy tells the robot how to move and when to invoke the sensor. The policy takes into account all the aspects of the problem: the performance of the sensor, the costs of sensing, probabilities of undesired events like losing the way or bumping into an obstacle etc. This is in contrast with previous approaches (for example [18, 15]) in which the highest localization accuracy possible was desired, but eventually compromised in order to account for the length of the path.

6 The POMDP Framework - A Brief Review

We will now describe the Partially Observable Markov Decision Process (POMDP) framework. We begin with the notion of Markov Decision Process (MDP).

Assume that an agent operates in a certain world. We assume that time is discrete and at each point in time the world is in a certain state s belonging to the set of possible states S. The agent chooses at each step an action a from the set of actions A. A reward function $R : S \times A \to \mathcal{R}$ determines the reward $R(s, a)$ the agent accepts when performing action a while the state is s. Apart from affecting the reward, the action a also affects the next state of the system as follows.

The state of the world in the next epoch is a random variable with a distribution which depends on both the current state s and the chosen action a. Let x_t denote the state at time t, and let a_t denote the action chosen at time t. We let $T(s, a, s')$ denote the state transition probabilities:

$$T(s, a, s') = \Pr(x_t = s' | x_{t-1} = s \text{ and } a_{t-1} = a)$$

Let $r_t = R(s_t, a_t)$ denote the reward collected at time t. The performance measure that is to be maximized is the sum of these rewards. The sum is either over a finite number of steps (so called finite horizon)

$$R = \sum_{t=1}^{N} r_t$$

or we may also look at an infinite horizon. In this case future rewards are discounted by a discount factor $0 < \lambda < 1$:

$$R = \sum_{t=1}^{\infty} \lambda^t r_t$$

In the Markov decision process (MDP) the agent always knows the state of the world x_t. The goal of the agent is to choose a policy or mapping between states and actions, which will maximize its expected total reward.

The POMDP framework is similar to the MDP framework except that the current state is unknown. Instead, a set O of possible observations exists, from which an observation o is given to the agent. The observation which the agent "measures" is a random variable which is distributed according to

$$\nu(o|s', a) = \Pr\left(\begin{array}{c} \text{after executing } a \text{ and reaching state} \\ s' \text{ the observation will be } o \end{array} \right)$$

Based on the dynamics of the system which are governed by $T(s, a, s')$, and on the measurements governed by $\nu(o|s', a)$, the agent maintains a "belief function" $b : S \to \mathcal{R}$. The belief function b is the probability distribution of the current state. The beliefs are propagated according to the dynamics and updated after each observation by using Bayes' rule.

A policy is a mapping from the belief state to actions: given the current belief b, what is the best action to choose ? As in the MDP, the goal is to find a policy which will maximize the expected total reward.

A more comprehensive introduction to POMDPs may be found in [9, 8].

In principle a POMDP is a particular case of MDP where the state space is the belief space. However the belief space is much too large (in fact infinite) for standard techniques for solving MDPs to be used. Therefore special algorithms have been proposed for POMDPs. Examples of such algorithms may be found in [9] and the references therein. However, even for problems with tens of states, finding the optimal policy exactly is computationally prohibitive. Therefore various researchers have worked on algorithms for finding approximations of the optimal policy. A review of approximation techniques and some new algorithms may be found in [8]. Other algorithms for finding approximate policies are presented in [13, 14, 19]. In the robotics context, a successful experience in which the search for the optimal policy was bypassed is reported in [17]. There actions were chosen according to the most probable action rule: at each state the correct action was defined, and the most probable action according to the current belief was chosen.

7 Description of the POMDP

We now turn to phrase our problem of rating navigation packages in terms of the POMDP framework. Let us assume that our environment is quantized to a grid. On the grid initial and goal configurations are marked. At each point on the grid the robot may choose one of six actions. It may either move in one of the four directions (up, right, down or left), or it may stop, or it may invoke its sensor to get an update of its position. Thus the state space S is the set of possible (discrete) locations, and the set of actions is

$$A = \{ \text{ up, right, down, left, update, stop } \}$$

After either a movement command or a "stop" command, no observation is given. We mark this as being given the "nil" observation. If the action taken was the "update" action, then the observation returned is a state $\hat{s}$, which is a random variable describing the result of invoking the sensor. The distribution of this observation varies with the true position the robot is in. This

way we model the variation in the performance of the sensor across the environment. Thus the set of observations is

$$O = \{ \text{ nil } \} \cup S$$

The uncertainty which develops in the position of the robot is governed by the dynamics of its movements. Let $T(s, a, s')$ denote the probability that performing the action a while being in s will lead the robot to s'. For $a \in \{ \text{ update, stop } \}$ we have

$$T(s, a, s') = \begin{cases} 1 & s = s' \\ 0 & s \neq s' \end{cases}$$

while for movement actions, $T(s, a, s')$ should model factors such as control inaccuracies.

We allow some states to be defined as occupied by obstacles. Bumping into an obstacle is modeled by allowing $T(s, a, s')$ to be possibly non-zero for an obstacle state s'. After hitting an obstacle s', the robot bounces back to free space.

We must now specify how different actions in different states are rewarded. The first step is to use the motion planner to generate the nominal paths from every (free) state to the goal state. This defines a *preferred* action at every free state - namely the first movement command according to the motion planner. We reward movement according to the motion plan with (say) 1 unit. Movement which is not according to the nominal motion plan is rewarded with a smaller $w < 1$ reward. Thus, the motion planner is incorporated into the POMDP by encouraging movements according to the motion plan. We will later see another way in which the motion planner affects the POMDP.

All other factors that are part of the navigation process should also affect the rewards in the POMDP. We will now consider them.

The cost of sensing is brought into account by the reward u given for the sensing action. This u may in general depend on the state s in which the sensing action is performed.

If the robot chooses to stop while it is in the goal position, a large reward g is collected. Stopping in non-goal positions results in a lower reward d.

To summarize, for a non-obstacle state s we have:

$$R_1(s, a) = \begin{cases} 1 & a \text{ is movement according to motion plan} \\ w & a \text{ is another movement} \\ u & a = \text{ update} \\ g & a = \text{stop and } s \text{ is the goal} \\ d & a = \text{stop and } s \text{ is not the goal} \end{cases}$$

In order to discourage bumping into obstacles, we give a (possibly negative) reward ob for any action performed while in an obstacle state:

$$R(s, a) = \begin{cases} R_1(s, a) & s \text{ is not an obstacle state} \\ ob & s \text{ is an obstacle state} \end{cases}$$

Choice of parameters: The different parameters above reflect the preferences one might have. For instance, if it is critical not to bump into obstacles then one has to lower the parameter ob. As another example, the cost of sensing is another parameter that is clearly dependent on the specific scenario one encounters. Although having a large number of parameters is undesirable, the POMDP has to express a large number of such different aspects of the problem. The POMDP framework is general enough to allow expression of these different factors, and this is done by choosing different rewards.

8 Discussion

In this part of the paper we have introduced navigation packages and have demonstrated that many interesting problems may be phrased as problems of rating navigation packages one with respect to the other. We have described a framework under which this problem may by treated. We conclude this part of the work with some additional remarks.

Introducing the Motion Planner As discussed earlier, many other researchers have worked on integrating the sensor performance map into the motion planning stage. The approach in most of these works was to strive for the highest possible sensing capability along the path, but compromise on this requirement in favor of path length. An underlying motion planner was not part of the approach. Instead, a motion planning algorithm was devised to find paths that optimize a criterion which includes both the path length and the sensing accuracy along the path.

We argue that a broader viewpoint must be used. Why should we strive for the highest possible sensing accuracy ? Instead, we strive for a *sufficient* sensing accuracy required for completing the nominal path. As a basis for the path we take a nominal path generated by a motion planner. This path is augmented with sensing operations. Unnecessary sensing will yield sub-optimal rewards in the POMDP and therefore will be discouraged.

We remark that Erdmann has addressed the issue of minimal sensing capability required by a motion plan in [6]. In that work, the robot does not need to know exactly at what state it is in, but only which action will advance it towards its goal. The distance from the goal is measured in terms of a nominal motion plan. It is argued that the sensor should be designed to answer this

question ("What action will now move me towards the goal ?") and in some sense this is the minimal sensing needed.

The Roles of the Motion Planner The nominal motion planner which is part of the navigation package serves as a "guiding force" in the POMDP. This is due to the preferred rewards being given to movement actions that are in accordance with the motion planner.

In addition, as is shown in the companion paper [1], the nominal motion planner has an important role in the search for an optimal policy. It gives rise to the initial value function from which we try to search for the optimal value function.

As is shown in the companion paper, the nominal motion planners may greatly affect the policy which is found for the POMDP. Thus, although the nominal motion plan is not used "as is", it still has an important role and it is certainly possible that one nominal motion plan would be much better to use than another.

In the next part of the paper [1] we present an algorithm which searches for the optimal policy in the POMDPs and some example applications.

Acknowledgment A. Adam and I. Shimshoni were supported in part by Israeli Ministry of Science Grants no. 9766 and no. 2104.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Towards a Meta Motion Planner B: Algorithm and Applications. Proceedings of ICRA 2001.

[2] A. Adam, E. Rivlin, and I. Shimshoni. Computing the sensory uncertainty field of a vision based localization sensor. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2993–2999, 2000.

[3] J. Borenstein, H.R. Everett, L. Feng, and D. Wehe. Mobile robot positioning: Sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, 1997.

[4] T. Celinski and B. McCarragher. Achieving efficient data fusion through integration of sensory perception control and sensor fusion. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1960–1965, 1999.

[5] T. Celinski and B. McCarragher. Improving sensory perception through predictive correction of monitoring errors. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2608–2613, 1999.

[6] Michael Erdmann. Understanding action and sensing by designing action-based sensors. *Int. Journal of Robotics Research*, 14(5):483–509, October 1995.

[7] T. Fraichard and R. Mermond. Path planning with uncertainty for car-like robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 27–32, 1998.

[8] M. Hauskrecht. *Planning and control in stochastic domains with imperfect information.* PhD thesis, MIT, 1997.

[9] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[10] A. Lambert and N. L. Fort-Piat. Safe actions and observation planning for mobile robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1341–1346, 1999.

[11] Jean-Claude Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, 1991.

[12] Anthony Lazanas and Jean-Claude Latombe. Motion planning with uncertainty: a landmark approach. *Artificial Intelligence*, 76:287–317, 1995.

[13] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *Proc. of 12'th Int. Conf. on Machine Learning*, pages 362–370, 1995.

[14] R. Parr and S. Russell. Approximating optimal policies for partially observable stochastic domains. In *Proc. of 14'th IJCAI*, 1995.

[15] N. Roy, W. Bugard, D. Fox, and S. Thrun. Coastal navigation - mobile robot navigation with uncertainty in dynamic environments. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 35–40, 1999.

[16] R. Sharma and H. Sutanto. A framework for robot motion planning with sensor constraints. *IEEE Transactions on Robotics and Automation*, 13(1):61–73, February 1997.

[17] R. Simmons and S. Koenig. Probabilistic navigation in partially observable environments. In *Proc. of 14'th IJCAI*, 1995.

[18] H. Takeda, C. Facchinetti, and J.C. Latombe.' Planning the motions of a mobile robot in a sensory uncertainty field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(10):1002–1017, October 1994.

[19] S. Thrun. Monte Carlo POMDPs. In *Proc. of NIPS*, 1999.