Dynamic programming with partial information to overcome navigational uncertainty in a nautical environment

Chris Beeler
Department of Mathematics and Statistics
University of Ottawa
Ottawa, Canada
christopher.beeler@uottawa.ca

Mark Crowley
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada

Isaac Tamblyn
Department of Physics
University of Ottawa
Ottawa, Canada
Vector Institute for Artificial Intelligence
Toronto, Canada
isaac.tamblyn@uottawa.ca

Xinkai Li
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Canada

Maia Fraser
Department of Mathematics and Statistics
University of Ottawa
Ottawa, Canada

Abstract—Using a toy nautical navigation environment, we show that dynamic programming can be used when only partial information about a partially observed Markov decision process (POMDP) is known. By incorporating uncertainty into our model, we show that navigation policies can be constructed that maintain safety. Adding controlled sensing methods, we show that these policies can also lower measurement costs at the same time.

Index Terms—Dynamic programming, Markov processes, Risk management

1. Introduction

Uncertainty creates a major obstacle in solving control problems. The goal of these problems is to construct a policy that is expected to produce optimal trajectories. In some cases, uncertainty only causes deviations from the optimal trajectory, which may still result in an acceptable solution. For example, if a driver is uncertain of exactly which road they are on, they might deviate from the optimal route to their destination; however, they can still arrive by a less optimal route. In other cases, small deviations can lead to highly undesired results. With the previous example, if a driver is instead uncertain of where they are on the road, this can result in a collision, which we refer to as a catastrophic failure. Even if these deviations are symmetric in nature, catastrophic failure could be the most likely result.

Markov Decision Processes (MDPs) [1] are a common class of control problems that are very well studied in both dynamic programming (DP) [2] and reinforcement learning (RL) [3] (both traditional [4] and deep [5]). While the majority of MDP results are simulated, there are real world applications. The Airborne Collision Avoidance System X [6] uses methods of solving MDPs with DP to aid actual operating aircraft avoid collisions in real time, using a distribution of estimates for the state of the surrounding aircraft. Formulating an MDP this way is known as a Partially Observed MDP (POMDP) [7]. While these problems are also well studied in DP [8], [9], it has been much more recent that they have been studied in RL [10], [11].

In an MDP, the state of the system is known, however, in a POMDP it must be estimated, leading to some amount of uncertainty. Much of the difficulty in solving a POMDP stems from estimating the state of the system before choosing an action. This is where the majority of research in this area focuses. Controlled sensing problems are a special type of POMDP that focuses on problems with actions that reduce uncertainty for a cost, rather than actions that modify the state of the system itself. Some work has been done in this area [12], [13], however, it is still a largely unexplored area of research. Very recently, work has been done using RL on the combination of controlled sensing and traditional POMDPs with Active Measure RL [14] and Action-Contingent Noiselessly Observable MDPs [15].

With DP methods, it is assumed that the parameters that
define the system are completely known. With RL methods, it is assumed no information of the system is known a priori but is accessible via experiences. This means that if only partial information of the system is known, DP cannot be used. While RL can be used, these methods do not take any knowledge of the system into account. The work presented here aims to bridge this gap by leveraging partial information of a system in the construction of solutions, specifically in the combined area of traditional and controlled sensing POMDPs that have both state modifying and uncertainty reducing actions.

The systems that this problem structure applies to include, but is not limited to: navigation [6], healthcare [16], and even chemical experiments. In a chemical experiment, there are many variables to consider and even slight variations in them can change the outcome of a reaction. While a chemist can record every step they have made throughout an experiment, there will always be variations in the outcome. The only way to determine this variation is to take various measurements, each with an associated cost. Hence the problem of optimally performing an experiment while managing access to various measurements exists in the combined space of traditional and controlled sensing POMDPs.

Nautical navigation has been the subject of several DP studies [17]–[20], however, the primary focus has been on collision avoidance and route optimization (i.e. speed and fuel consumption) rather than uncertainty and controlled sensing. Here we introduce a toy nautical navigation environment described in detail in Sec. 3. We assume only partial information of the system is available to the agent, leading to a level of uncertainty, and a set of information revealing actions (or measurements) are accessible that help reduce uncertainty at a cost.

2. Background

An MDP is a Markov chain where an agent controls the transition probabilities of moving from state to state during a trajectory. The state \( s \in \mathcal{S} \) describes the system at any given moment. Typically, an action \( a \in \mathcal{A} \) controls the state-to-state probability transition matrix \( P(a) \), where \( P_{ij}(a) \) is the probability that the system will enter state \( j \) given action \( a \) is taken at the present state \( i \). These are the state-modifying actions. The cost function \( c(s, a) \) measures the expected cost of performing action \( a \) at state \( s \). Lastly, the discount factor \( 0 \leq \rho < 1 \), measures how important the expected future costs should affect choosing an action. This gives us a formal MDP descriptor quintuple:

\[
(\mathcal{S}, \mathcal{A}, P(a), c(s, a), \rho) .
\] (1)

When solving an MDP, the goal is to find a policy \( \mu : \mathcal{S} \rightarrow \mathcal{A} \) that minimizes the expected total cost incurred. An optimal policy \( \mu^* \) is one that incurs the global minimum expected total cost when employed. This also requires the value function \( V : \mathcal{S} \rightarrow \mathbb{R} \), where the value represents how optimal any given state is, i.e. lower expected total costs are more optimal. When the MDP quintuple is known completely, they are both found by Bellman’s DP algorithm [2]:

\[
V_n(s) = \min_{a \in \mathcal{A}} Q_n(s, a)
\]

\[
\mu_n^*(s) = \arg\min_{a \in \mathcal{A}} Q_n(s, a),
\] (2)

where the Q-function is defined as

\[
Q_n(s, a) = c(s, a) + \rho \sum_{s' \in \mathcal{S}} P_{ss'}(a)V_{n-1}(s'),
\] (3)

where \( V_0 \equiv 0 \). Taking the limit as \( n \rightarrow \infty \) gives the optimal value function and policy.

Similar to MDPs, we define a POMDP as a hidden Markov model where an agent controls the transition probabilities of moving from state to state. As the state is not directly observable in a POMDP, we have, in addition to the previous components, an observation \( o \in \mathcal{O} \) is what the agent observes based on the probability function \( B(a) \), where \( B_{ij}(a) \) is the probability that observation \( j \) will occur given action \( a \) is taken at the present state \( i \). This gives us a formal POMDP descriptor septuple:

\[
(\mathcal{S}, \mathcal{A}, \mathcal{O}, P(a), B(a), c(s, a), \rho) .
\] (4)

When this septuple is known completely, the probability distribution of which states the system could be in at time \( t \), or belief state \( \pi_t \), is given by the filter

\[
T(\pi_{t-1}, o, a) = \frac{\text{diag}(B_{so}|s \in \mathcal{S})P_T(a_{t-1})\pi_{t-1}}{\sigma(\pi, o, a)},
\] (5)

where

\[
\sigma(\pi, o, a) = 1^T_\mathcal{S} \text{diag}(B_{so}|s \in \mathcal{S})P_T(a)\pi,
\] (6)

and \( \pi_t(i) \) is the probability that state \( i \) is the actual state of the system at time \( t \). The optimal function and policy are then found again by the modified Bellman’s DP algorithm [7]:

\[
V_n(\pi) = \min_{a \in \mathcal{A}} Q_n(\pi, a)
\]

\[
\mu_n^*(\pi) = \arg\min_{a \in \mathcal{A}} Q_n(\pi, a),
\] (7)

where the Q-function is now defined as

\[
Q_n(\pi, a) = \sum_{s \in \mathcal{S}} c(s, a)\pi(s)
+ \rho \sum_{o \in \mathcal{O}} V_{n-1}(T(\pi, o, a))\sigma(\pi, o, a),
\] (8)

where \( V_0 \equiv 0 \).

In a controlled sensing POMDP, the state transition matrix is typically independent of the chosen action, but the observation and cost functions may not be. Note that measurement actions that can be taken without the state changing, cause the transition matrix to become the identity for that action. This is equivalent to time not advancing during this step.

When solving a POMDP with RL, the main difference is that it is assumed that \( P(a), B(a), \) and \( c(s, a) \) are not known at all. If these elements are partially known, that
information is ignored. In the next section, we present a toy environment in which we have partial information and controlled sensing actions.

3. Nautical Navigation Environment

To explore the concept of partial information POMDPs, we introduce a toy nautical navigation environment. In this environment, the agent must navigate a submarine through a set of islands to a specified circular target region. To navigate, the agent must specify a heading and throttle setting that provides a movement vector, shown in Fig 1(a). Typically there is a non-linear relation between throttle and speed. In this case, speed through the water is the square root of throttle, as shown in Fig. 1(b). An RL agent would have to learn this relationship, however, in our case, it can be included in our partial information setup. If the agent reaches the target region, it receives a negative cost (also called a reward) and if the agent crashes into an island, it receives a large positive cost. The trajectory terminates when either of these cases occurs.

This system also contains water currents that cause drifts from the expected trajectory of the specified movement vector. Together, the movement vector and water current give the velocity of the submarine over land, defining how the state of the system changes. Unlike the set of island obstacles, the exact water current is assumed to be unknown by the agent, (but it can be partially observed indirectly). If the agent knows the movement vector chosen and their true position before and after an action, the average water current over that action can be obtained from the displacement between the expected and true final positions, as shown in Fig. 1(c).

The unknown water current gives rise to a level of uncertainty in the movement of the submarine, which in turn, gives rise to a level of uncertainty in the resulting position. To help the agent overcome these uncertainties, we introduce two measurement actions:

1) **GPS**: Returns the true position of the submarine, therefore reducing positional uncertainty to zero. This allows for the calculation of the average water current between the previous and present GPS measurements. Hence, this measurement slightly reduces positional uncertainty, although not completely.

2) **Current Profiler**: Returns the true water current for the true position of the submarine, therefore reducing the water current uncertainty to zero. Note that because the analytic water current is unknown, this measurement does not reveal any positional information regarding the position of the submarine. Hence, the positional uncertainty is unaffected.

Note that for these measurement actions, $P(a) = I$ and $c_m(a) \equiv \text{const.}$ where the constant is some specific instantaneous measurement cost assigned for employing that measurement and $c_m(a) = 0$ for all non-measurement actions. These costs represent both the monetary costs of using and maintaining each device and the time required to operate them.

3.1. Charts

The system the agent is navigating in is contained inside a rectangular area with periodic boundary conditions and dimensions $x_{\text{max}}$ and $y_{\text{max}}$. We call the pairing of this area with the set of islands a chart. Each island obstacle is represented by a 2-dimensional Gaussian function where the parameters are independently sampled uniformly from their respective ranges described in Sec. S1. We then define the land function $f(x, y)$ as the summation over several islands. The notation used here is: 0 is the ocean floor, 1 is sea level, and 0.9 is the height at which the submarine operates, i.e. the
agent navigating to any point \((x, y)\) such that \(f(x, y) \geq 0.9\) results in a crash. During a trajectory, the agent always has access to the charts.

### 3.2. Water Currents

While it is assumed the agent does not know the analytic water current, it is generated deterministically for each given land function. The water current vector \(W(x, y)\) at \((x, y)\) is perpendicular to \(\nabla f(x, y)\) with magnitude bounded by \(w_{\text{max}}\) and linearly related to \(-\|\nabla f(x, y)\|_2\), with the specifics presented in Sec. S2.

With the water current formally defined, we can now define the velocity of the submarine at any given time. For a specified movement vector \(M\), the velocity of the submarine is given by

\[
v(x, y) = \frac{M}{\|M\|_2} + W(x, y).
\]

### 4. Finding an Optimal Policy

With a navigation environment defined, we can now develop a policy construction method. As the agent does not have complete knowledge of the system, Bellman’s DP algorithms presented in Sec. 2 cannot be used directly. In this system, if there are no water currents, or if the agent knows the water currents exactly, the problem becomes an MDP and Eq. 2 is applicable. As we assume the agent does not know the water current, we turn to the former to be the base model for constructing a solution. In the next section, we present how to form this base.

#### 4.1. Value Function

During a single trajectory, \(f(x, y)\) and \(W(x, y)\) do not change, therefore for simplicity we refer to the submarine position \((x, y)\) as the state of the system. The velocity of the submarine need not be included as we assume the time scale of acceleration and changing directions insignificant relative to the time between actions. In the first step in constructing our solution, we assume the system contains no water current, i.e. \(W(x, y) \equiv 0\). With this assumption, for any given chart we can generate a value function using Eq. 2, where \(\|M\|_2 \leq 1\). To encourage faster routes, we introduce a fuel cost defined as \(c_f(a) = 0.01\|M\|_2\) for all non-measurement actions. We define the positional cost function as \(c_p(x, y) = 100\) for any \((x, y)\) such that \(f(x, y) \geq 0.9\), \(c_p(x, y) = -1\) for any resulting submarine positions \((x, y)\) inside the specified target region such that \(f(x, y) < 0.9\), and \(c_p(x, y) = 0\) otherwise. This gives us the cost function

\[
c(x, y, a) = c_p(x, y) + c_f(a) + c_m(a). \tag{10}
\]
Note that the trajectory terminates when $c_p(x, y) \neq 0$.

For any chart, with or without water currents, we have $V_0 \equiv 0$ and $V_1(x, y) = \min_{a \in A} c(x, y, a)$. Examples of a chart without and with water currents are shown in Figs. 2(a) & 2(b) with $V_1$ for each case shown in Figs. 2(c) & 2(d) respectively. In this system, Eq. 2 results in a converged value function $V$ after finite $n$. The converged value functions for the examples above are shown in Figs. 2(e) & 2(f) respectively, with three regions of notable difference between the two circled. As we assume the agent does not know $W(x, y)$, we continue with the value functions of the type in Fig. 2(e) for the next section.

4.2. Policy Construction

With the water current unknown, the goal is to construct a policy in a similar manner to Bellman’s DP algorithm for POMDPs shown in Eq. 7. Doing so requires using the expected states and uncertainty to determine the agent’s belief state. At the initial step of each trajectory, the agent knows the true initial submarine position and water current for that specific position, therefore uncertainty in both is zero. This is the first of four cases considered. As the water current changes when the submarine moves away from this position, uncertainty in water current grows during any non-measurement action taken, leading to the growth of uncertainty in position shown in Fig. 3(a). With the expected trajectory based on the known position, starting water current, and action taken, this gives us a distribution of trajectories that may occur. Therefore each possible non-measurement action can be assigned an expected value and instantaneous cost based on these distributions. Then the policy becomes: select the non-measurement action with the lowest expected value (i.e. lowest expected total cost) based on the distribution of trajectories.

During all subsequent steps of the trajectory, the agent has an expected position and water current, however, it also has uncertainty in both. This is the second case. As before, uncertainty in position increases due to uncertainty in the water current growing throughout an action. However, uncertainty in position also increases due to the initial non-zero uncertainty in water current. This combination leads to the growth of uncertainty in position shown in Fig. 3(b), where the initial positional uncertainty is now non-zero. As before, this gives us a distribution of trajectories that may occur. Hence each possible non-measurement action can be assigned an expected value and instantaneous cost. As mentioned before, the agent has access to two types of measurements to reduce this uncertainty; each with an associated instantaneous cost. If the lowest expected instantaneous cost of any action is greater than the cost of any of the available measurement actions, the policy becomes: take a measurement. If the expected position of the submarine is inside the target region, the policy becomes: specifically take a GPS measurement. Otherwise, the policy becomes: select the non-measurement action with the lowest expected value based on the distribution of trajectories.

If the GPS measurement is taken, the positional uncertainty goes back to zero and the water current uncertainty is slightly reduced; however, it is still non-zero. This is the third case. During any non-measurement action now, the positional uncertainty grows similar to the second case, with however, an initial positional uncertainty of zero, shown in Fig. 3(c).

If the current profiler measurement is taken, the water current uncertainty goes back to zero and the positional uncertainty is unaffected, therefore still non-zero. This is the fourth case. During any non-measurement action now, the positional uncertainty grows similarly to the first case, with however, an initial positional uncertainty of non-zero, shown in Fig. 3(d).

In either the third or fourth cases, the expected values and instantaneous costs must be re-determined for each non-measurement action. If the lowest expected cost is greater than the cost of the other measurement, that measurement will also be taken, bringing the agent back to the first case. Otherwise, the policy becomes: select the non-measurement action with the lowest expected value based on the new distribution of trajectories.
5. Computational Set-up

For computational purposes, we discretize the action space to 96 non-measurement actions for the agent (6 throttle settings and 16 heading directions) and state-space to a resolution of $152 \times 152$ for the value function. Note that only the input to the value function is discretized and the actual state-space remains continuous. As the relationship between speed through the water and throttle is included in the partial information, the discrete actions available are chosen such that the non-measurement action choices are linear with respect to speed through the water for simplicity, as shown in Fig. 1(b), inclusive of 0 and 1.

For the chart generation, we have $x_{\text{max}} = y_{\text{max}} = 10$ for all charts and a varying number of islands $0 \leq N \leq 20$. We consider $1 \leq N \leq 5$ charts of low island density, $8 \leq N \leq 12$ charts of medium island density, and $16 \leq N \leq 20$ charts of high island density. 50 charts from each density group will be used with 10 different initial states each.

For the water currents, we have $w_{\text{max}} = 0.5$. The linear rates at which the uncertainty grows will be parameters of experimentation, varying from 0 to $2w_{\text{max}}$, but the estimates in water currents are bounded by $\|\hat{W}(x, y)\|_2 \leq w_{\text{max}}$. The GPS measurement has a cost of 0.45 and the current profiler measurement has a cost of 0.1.

6. Results

Based on preliminary tests, any policy that does not reach the target or crash within 20 steps is highly unlikely to do either. For this reason, we limit all trajectories to 20 steps. We consider the following three types of outcomes: a policy that reaches the target within 20 steps is considered successful, a policy that crashes within 20 steps is a failure, and a policy that does neither is unsuccessful but not a failure.

For each chart, a value function is generated using the method described in Sec. 4.1. For each initial state, a policy is constructed several times using the method described in Sec. 4.2, where the uncertainty growth rate is varied. An uncertainty rate of zero is equivalent to assuming no water currents exist in the system.

Fig. 4(a) shows the probability of success as a function of uncertainty growth rate for policies constructed for low, medium, and high island densities. With an uncertainty growth rate of zero, the probability of success is only 33.0%, 29.2%, and 30.8% for low, medium, and high island densities respectively, however, even increasing this parameter to only 0.05 doubles these on average. The maximum probabilities of success of 82.1%, 86.2%, and 81.8% occur at uncertainty growth rates of 0.79, 0.68, and 0.68 for low, medium, and high island densities respectively, however, even increasing this parameter to only 0.05 doubles these on average. The maximum probabilities of success of 82.1%, 86.2%, and 81.8% occur at uncertainty growth rates of 0.79, 0.68, and 0.68 for low, medium, and high island densities respectively. These probabilities steadily decrease if the uncertainty growth rate is increased further as any target remotely close to an island becomes too risky to reach, (i.e. the expected cost of crashing with these parameters is greater than the cost of success).
Fig. 4(b) instead shows the probability of not crashing as a function of uncertainty growth rate with similar behavior as before. As expected, increasing the density of islands increases the probability of crashing, however, we achieve a 0% probability of crashing for low and medium island densities and 1.3% with a high island density. Unlike before, the probability of crashing stays relatively low as the uncertainty growth rate goes to 1.0. Policies in that case end up choosing actions that have the least likelihood of crashing rather than trying to reach the target.

Fig. 4(c) shows the average number of measurements taken per step as a function of uncertainty growth rate. Policies with a low island density measure fewer times on average, as expected. Policies with medium or high island densities still measure less than once per action on average, but with much more variance.

7. Conclusion

Motivated by real world applicability of POMDPs and systems with uncertainty, we have shown that partial information can be leveraged with DP methods to construct navigational policies that both maintain safety and reduce total measurement cost. The toy navigation environment we introduced serves as a relevant introduction to the problems of interest in the combined area of traditional and controlled sensing POMDPs. The methods provided allow the construction of value functions through DP that contain the basic information of the system of interest.

We show that without any additional constraints, the policies produced using these value functions perform very poorly. However, when uncertainty methods are included, the probability of success on average is doubled and the probability of crashing is brought to (or nearly to) zero. We also show that these policies are able to reduce the number of measurements taken in order to navigate to a specified target region.

Acknowledgment

C.B. performed work at the National Research Council of Canada under the AI4D program. I.T. acknowledges NSERC. The code used in this study can be found at https://clean.energyscience.ca/gyms.

References

[1] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[2] R. Bellman, “The theory of dynamic programming,” *Bulletin of the American Mathematical Society*, vol. 60, no. 6, pp. 503–515, 1954.

[3] R. S. Sutton and A. G. Barto, “Reinforcement learning: An introduction,” 2018.

[4] R. S. Sutton, D. Precup, and S. Singh, “Between mdp and semi-mdps: A framework for temporal abstraction in reinforcement learning.” *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., “Human-level control through deep reinforcement learning,” *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[6] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos, “Next-generation airborne collision avoidance system,” tech. rep., Massachusetts Institute of Technology-Lincoln Laboratory Lexington United States, 2012.

[7] V. Krishnamurthy, *Partially observed Markov decision processes*. Cambridge university press, 2016.

[8] S. Seuken and S. Zilberstein, “Memory-bounded dynamic programming for dec-pomdps,” in *IJCAI*, pp. 2009–2015, 2007.

[9] K. G. Papakonstantinou and M. Shinozuka, “Planning structural inspection and maintenance policies via dynamic programming and markov processes. part ii: Pomdp implementation,” *Reliability Engineering & System Safety*, vol. 130, pp. 214–224, 2014.

[10] G. Singh, S. Peri, J. Kim, H. Kim, and S. Ahn, “Structured world belief for reinforcement learning in pomdp,” in *International Conference on Machine Learning*, pp. 9744–9755, PMLR, 2021.

[11] K. Azizzadenesheli, A. Lazaric, and A. Anandkumar, “Reinforcement learning of pomdps using spectral methods,” in *Conference on Learning Theory*, pp. 193–256, PMLR, 2016.

[12] V. Krishnamurthy, “Convex stochastic dominance in bayesian localization, filtering, and controlled sensing pomdps,” *IEEE Transactions on Information Theory*, vol. 66, no. 5, pp. 3187–3201, 2019.

[13] S. Bhatt, V. Krishnamurthy, and M. Rangaswamy, “Controlled sentiment sampling for information fusion in social networks,” in *2018 21st International Conference on Information Fusion (FUSION)*, pp. 1157–1162, IEEE, 2018.

[14] C. Bellinger, R. Coles, M. Crowley, and I. Tamblyn, “Active measure reinforcement learning for observation cost minimization: A framework for minimizing measurement costs in reinforcement learning,” in *Canadian Conference on Artificial Intelligence*, Lecture Notes in Computer Science (LNCS), p. 12, Springer, Springer, 2021.

[15] H. Nam, S. Fleming, and E. Brunskill, “Reinforcement learning with state observation costs in action- contingent noiselessly observable markov decision processes,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[16] A. K. Jha, D. C. Chan, A. B. Ridgway, C. Franz, and D. W. Bates, “Improving safety and eliminating redundant tests: cutting costs in us hospitals,” *Health affairs*, vol. 28, no. 5, pp. 1475–1484, 2009.

[17] R. Zaccone, E. Ottaviani, M. Figari, and M. Altosole, “Ship voyage optimization for safe and energy-efficient navigation: A dynamic programming approach,” *Ocean engineering*, vol. 153, pp. 215–224, 2018.

[18] X. Geng, Y. Wang, P. Wang, and B. Zhang, “Motion plan of maritime autonomous surface ships by dynamic programming for collision avoidance and speed optimization,” *Sensors*, vol. 19, no. 2, p. 434, 2019.

[19] S. Wei and P. Zhou, “Development of a 3d dynamic programming method for weather routing,” *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 6, no. 1, pp. 79–85, 2012.

[20] A. Fan, Z. Wang, L. Yang, J. Wang, and N. Vladimir, “Multi-stage decision-making method for ship speed optimisation considering inland navigational environment,” *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 235, no. 2, pp. 372–382, 2021.
Supplementary Information for “Dynamic programming with partial information to overcome navigational uncertainty in a nautical environment”

Chris Beeler  
Department of Mathematics and Statistics  
University of Ottawa  
Ottawa, Canada  
christopher.beeler@uottawa.ca

Mark Crowley  
Department of Electrical and Computer Engineering  
University of Waterloo  
Waterloo, Canada

Isaac Tamblyn  
Department of Physics  
University of Ottawa  
Ottawa, Canada  
Vector Institute for Artificial Intelligence  
Toronto, Canada  
isaac.tamblyn@uottawa.ca

Xinkai Li  
Department of Electrical and Computer Engineering  
University of Waterloo  
Waterloo, Canada

Maia Fraser  
Department of Mathematics and Statistics  
University of Ottawa  
Ottawa, Canada

Abstract—Using a toy nautical navigation environment, we show that dynamic programming can be used when only partial information about a partially observed Markov decision process (POMDP) is known. By incorporating uncertainty into our model, we show that navigation policies can be constructed that maintain safety. Adding controlled sensing methods, we show that these policies can also lower measurement costs at the same time.

Index Terms—Dynamic programming, Markov processes, Risk management

1. Chart Generation Method

The system the agent is navigating in is contained inside a rectangular area with periodic boundary conditions and dimensions \(x_{\text{max}}\) and \(y_{\text{max}}\). Each these island obstacles is represented by a 2-dimensional Gaussian function, defined as

\[
g(x, y) = Ae^{-(a(x-x_0)^2+2b(x-x_0)(y-y_0)+c(y-y_0)^2)} \tag{1}
\]

where \(x_0 \in [0, x_{\text{max}})\), \(y_0 \in [0, y_{\text{max}})\), \(A \in [1, 2]\), \(a, c \in [1, \infty)\) are independently sampled from uniformly from their respective ranges and \(b \in (-\sqrt{ac}, \sqrt{ac})\) is also sampled uniformly, (however it is dependent on \(a\) and \(c\)). We then define the land function as

\[
f(x, y) = \sum_{i=1}^{N} \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} g_i(x + jx_{\text{max}}, y + ky_{\text{max}}), \tag{2}
\]

where \(N\) is the number of islands and the parameters for each \(g_i\) are sampled as described above and independently from each other island. While true periodic boundary conditions require the infinite sums, the bounds \(-1 \leq j, k \leq 1\) are sufficient for our purposes.

2. Water Current Generation Method

For each island \(g_i(x, y)\), the water current \(W(x, y)\) vector at position \((x, y)\) has direction given by

\[
w(x, y) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \sum_{i=1}^{N} \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \left((-1)^{m_i(x, y)} \times \nabla g_i(x + jx_{\text{max}}, y + ky_{\text{max}})\right), \tag{3}
\]

where each \(m_i\) only returns the discrete values 0 and 1 and is chosen to maximize \(\|w(x, y)\|_2\). The water current \(W(x, y)\) function is then defined as

\[
W(x, y) = \frac{w_{\text{max}} - \|w(x, y)\|_2}{2w_{\text{max}}\|w(x, y)\|_2}w(x, y) \tag{4}
\]
if \( f(x, y) < 0.9 \) and \( 0 < \|w(x, y)\|_2 \leq w_{\text{max}} \), \( W(x, y) = 0 \)

if \( f(x, y) \geq 0.9 \) or \( \|w(x, y)\|_2 > w_{\text{max}} \), and

\[
W(x, y) = \frac{w_{\text{max}}}{\|w(x, y)\|_2} w(x, y)
\]  

(5)

if \( f(x, y) < 0.9 \) and \( w(x, y) = 0 \), where \( w_{\text{max}} \geq 0 \). In other words, the water current vector at \((x, y)\) is perpendicular to \( \nabla f(x, y) \) with magnitude bounded and related to \(-\|\nabla f(x, y)\|_2\).

**Acknowledgment**

C.B. performed work at the National Research Council of Canada under the AI4D program. I.T. acknowledges NSERC.