Dynamic programming with incomplete information to overcome navigational uncertainty in POMDPs

Chris Beeler[1,2,*], Xinkai Li[3], Colin Bellinger[2], Mark Crowley[3], Maia Fraser[1], Isaac Tamblyn[4,5]
[1] Department of Mathematics and Statistics, University of Ottawa
[2] Digital Technologies, National Research Council of Canada
[3] Department of Electrical Engineering, University of Waterloo
[4] Department of Physics, University of Ottawa
[5] Vector Institute for Artificial Intelligence

Abstract

Using a generalizable novel nautical navigation environment, we show how dynamic programming can be used when only incomplete 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, outperforming the baseline performance of traditional dynamic programming for Markov decision processes (MDPs). Adding in controlled sensing methods, we show that these policies can also lower measurement costs at the same time.

Keywords: Dynamic programming, Partially observable, Markov decision processes, Risk management, Controlled sensing

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 causes small deviations from the optimal trajectory, which are nevertheless still acceptable solutions. 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 via a less optimal route. In other cases, uncertainty 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) [2–5] and reinforcement learning (RL) [6] (both traditional [7–9] and deep [10–12]). While the majority of MDP results are simulated, there are real-world applications. The Airborne Collision Avoidance System X [13] uses methods of solving MDPs with DP to aid actual operating aircraft to avoid collisions in real-time, using a distribution of estimates for the state of the surrounding aircraft. We will study problems like this through the formalism of Partially Observed MDP (POMDP) [14] which we describe below and use to present a modified version of Bellman's dynamic programming equations, Equation (4.2). While POMDPs are also well studied in DP [15–19], it is only more recently that they have been studied in RL [20–23].

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 where some of the actions reduce uncertainty for a cost, rather than modifying the state of

[*] christopher.beeler@uottawa.ca

the system. Some work has been done in this area [14, 24–28], however, it is still largely unexplored.

Separate from the question of the partial observation of the current state is knowledge of the environment itself, i.e. the space of all possible states and how the available actions cause transitions between them. Depending on how much of the system's information is available to the agent, different approaches are possible to optimize agent behaviour. DP methods require full knowledge of the environment and thus amount strictly to optimization, without "learning" per se. At the other end of the spectrum, RL methods assume little or no access to information about the system; they involve learning from experience to deduce which actions have the most desirable effects. In this work, we consider POMDPs whose underlying MDP is fully known to the agent. The MDP setting allows for analytic solutions via DP, and we propose a method to adapt such solutions to the related POMDP where the agent must contend with uncertainty regarding its current state. While purely RL methods could be used instead, they would not take into account the agent's knowledge of the MDP. Our work thus fills a gap, providing POMDP solutions in a DP-grounded rather than RL-grounded approach. In particular, the settings we consider include the areas of controlled sensing and traditional POMDPs.

The systems that this problem structure applies to include, but are not limited to: navigation [13], healthcare [29], and even chemical experiments [30]. 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 is located in the combined space of traditional and controlled sensing POMDPs.

Nautical navigation has been the subject of several DP studies [31–33], 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 nautical navigation environment described in detail in Section 3. We assume the agent has incomplete access to the information of the system, which leads to a level of uncertainty. A set of information-revealing actions (or measurements) are accessible that help reduce uncertainty at a cost.

The main contributions we present here are: a novel nautical navigation environment that allows for the control of the level of information and can be generalized to many fields, a modified version of Bellman's dynamic programming equations, Equation (4.2), and policy construction for POMDPs with incomplete information, as well as POMDP solutions that combine state altering actions with controlled sensing techniques that outperform the baseline of non-adapted dynamic programming solution for the underlying MDP.

2. Background

An MDP is a paradigm consisting of an agent and an environment. The agent can interact with the environment by taking actions that cause a transition from one state to another, incurring a cost (also known as a negative reward) for that transition. The goal of the agent is to minimize the cumulative cost, where cumulative cost can be defined in various ways. Formally, a finite MDP is defined by the quintuple: $(\mathcal{S}, \mathcal{A}, P, c, \gamma)$, where $\mathcal{S}$ is the state space, $|S| = n$, $\mathcal{A}$ is the action space, $P : \mathcal{A} \to \mathbb{R}^{n\times n}$ is the function of state-to-state transition probability matrices, c is the cost function with $c(s, a) = \mathbb{E}_{s'\in\mathcal{S}} c(s, a, s')$, and $\gamma \in [0, 1)$ is the discount factor which measures how important the expected future costs are when choosing an action. For given states $i, j \in \mathcal{S}$ and action $a \in \mathcal{A}$, $P_{ij}(a)$ is the probability that the system will enter state j given action a is taken at the present state i and $c(i,a,j)$ is the cost incurred by the agent by transitioning from state i to j with action a.

When solving an MDP, the goal is to find a policy $\mu : \mathcal{S} \to \mathcal{A}$ that minimizes the expected total cost incurred. In fact, an MDP combined with a policy forms a Markov chain with costs associated with transitions. An optimal policy μ^* is one that incurs the global minimum expected cumulative cost when employed, where the global minimum is over all possible policies. This can be expressed in terms of the value function $V : \mathcal{S} \to \mathbb{R}$, where the value of a state is the minimum expected cumulative cost at said state. The value function represents how optimal any given state is, i.e. lower valued states are more optimal as they have lower expected cumulative costs. When the MDP tuple is known completely, the value function and optimal policy are both found by Bellman's DP algorithm [2]: $V_n(s) = \min_{a\in\mathcal{A}} \mathcal{Q}_n(s,a)$ and $\mu_n^*(s) = \operatorname{argmin}_{a\in\mathcal{A}} \mathcal{Q}_n(s,a)$, where the Q-function is defined as

$$\mathcal{Q}_n(s,a) = c(s,a) + \gamma \sum_{s'\in\mathcal{S}} P_{ss'}(a) V_{n-1}(s'), \tag{2.1}$$

where $V_0 \equiv 0$. Taking the limit as $n \to \infty$ gives the optimal value function and policy. An agent's goal is to find a policy that minimizes the value function over all possible states.

Similar to MDPs, the goal of an agent in a POMDP is to minimize the cumulative cost. Unlike in an MDP, the agent is not always able to directly observe the state of the environment. Formally, a POMDP is defined by the septuple: $(\mathcal{S}, \mathcal{A}, \mathcal{O}, P, B, c, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$, P, c, and γ are defined the same as for MDPs, $\mathcal{O}$ is the observation space, and B observation probability function. For a given state $i \in \mathcal{S}$, observation $j \in \mathcal{O}$, and action $a \in \mathcal{A}$, $B_{ij}(a)$ is the probability that the agent will observe j given that state i occurred after taking action a.

When this tuple is known completely, the belief state, π_t, is a probability distribution over $\mathcal{S}$ and is updated by

$$\pi_t = \frac{\operatorname{diag}(B_{so_t}(a_{t-1})|s \in \mathcal{S}) P^T(a_{t-1}) \pi_{t-1}}{\sigma(\pi_{t-1}, o, a)}, \tag{2.2}$$

where $\sigma(\pi, o, a) = \mathbf{1}_{\mathcal{S}}^T \operatorname{diag}(B_{so_t}|s \in \mathcal{S}) P^T(a)\pi$ [14]. Similarly, a policy for POMDPs is a map from this distribution on $\mathcal{S}$ to $\mathcal{A}$. The optimal function and policy are then found again by the modified Bellman's DP algorithm [14] where $V_n(\pi)$ is used instead of $V_n(s)$ and the Q-function is now defined as

$$\mathcal{Q}_n(\pi,a) = \sum_{s\in\mathcal{S}} c(s,a)\pi(s) + \gamma \sum_{o\in\mathcal{O}} V_{n-1}(T(\pi,o,a))\sigma(\pi,o,a), \tag{2.3}$$

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.

The above methods for solving POMDPs use DP and assume the agent has complete access to each element of the POMDP septuple. When solving a POMDP with RL, the main difference from the DP solutions is that it is assumed that P, B, and c are not available to the agent and must be learned. If the agent has incomplete access to this information, it is ignored in RL algorithms. In the next section, we present a novel environment in which the agent has incomplete access to information and controlled sensing actions.

3. Nautical Navigation Environment

To explore the concept of incomplete access to information POMDPs, we introduce a nautical navigation environment, that serves as an easy-to-understand and very generalizable system for our discussion. 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 Figure 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 Figure 1(b). An RL agent would have to learn this relationship, however, in our case, it can be included in our agent's information access setup. If the agent reaches the target region, it receives a negative cost (also known as 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.

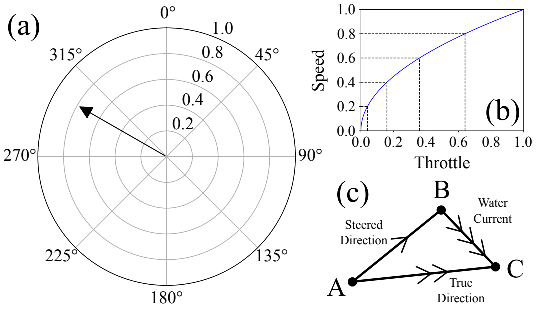

Figure 1. (a) A graphical representation of the non-measurement action space the agent can choose from at each step. (b) The relationship of throttle and speed through the water where the black dashed lines represent the throttle discretization used in Section 5. (c) A graphical representation of how the non-measurement actions map to the submarine's displacement. **A** represents the starting position of the submarine. **B** represents the estimated final position of the submarine based on the non-measurement action chosen by the agent (or velocity through the water) represented by the single arrow line. **C** represents the true final position of the submarine where the true path represented by the double arrow line is the combination of **B** and the water current represented by the triple arrow line (or velocity overland).

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 Figure 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. The agent has two measurement actions available to it, and can use them to help overcome these uncertainties:

(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 the water current 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 information regarding the position of the submarine. Hence, the positional uncertainty is unaffected.

Note that for these measurement actions, $P(a) = I$ and $c_{\mathrm{m}}(a) \equiv \text{const.}$ where the constant is some specific instantaneous measurement cost assigned for employing that measurement and $c_{\mathrm{m}}(a) \equiv 0$ 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.

Charts: The system the agent is navigating in is contained inside a rectangular area with periodic boundary conditions and dimensions $x_{\max}$ and $y_{\max}$. 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 to ensure

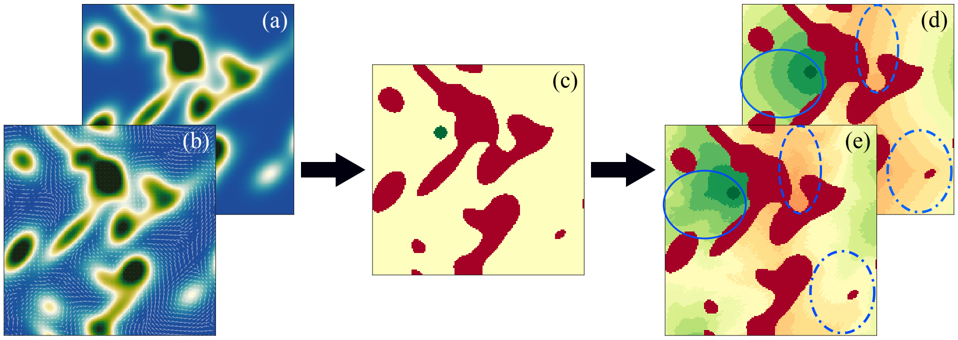

Figure 2. A graphical representation of the process of generating a value function without and with water currents. (a) Shows a chart without water currents and (b) shows the same chart with water currents. (c) Shows V_1 with a specified circular target region for the charts with and without water currents. (d) Shows the optimized value function for the chart without water currents and (e) shows the optimized value function for the chart with water currents. Regions of notable difference between (d) & (e) are highlighted with blue ellipsis, where each line style corresponds to a specific region.

convergence. We then define the land height function $f(x, y)$ as the summation over several islands. The notation for outputs of $f(x, y)$ used here are: 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.

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 $w_{\max}$ and linearly related to $-\|\nabla f(x, y)\|_2$.

4. **Finding an Optimal Policy**

With a navigation environment defined, we can now use it as an example of how to develop a policy construction method. As the agent does not have complete knowledge of the system, Bellman's DP algorithms presented in Section 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 Bellman's equation 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 is 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 Bellman's equation, where $\|M\|_2 \leq 1$. To encourage faster routes, we introduce a fuel cost defined as $c_{\mathrm{f}}(a) = 0.01\|M\|_2$ for all non-measurement actions. We define the positional cost function as $c_{\mathrm{p}}(x, y) = 100$ for any (x, y) such that $f(x, y) \geq 0.9$, $c_{\mathrm{p}}(x, y) = -1$ for any resulting submarine positions (x, y) inside the specified target region such that $f(x, y) < 0.9$, and $c_{\mathrm{p}}(x, y) = 0$ otherwise. This gives us the cost function $c(x, y, a) = c_{\mathrm{p}}(x, y) + c_{\mathrm{f}}(a) + c_{\mathrm{m}}(a)$,

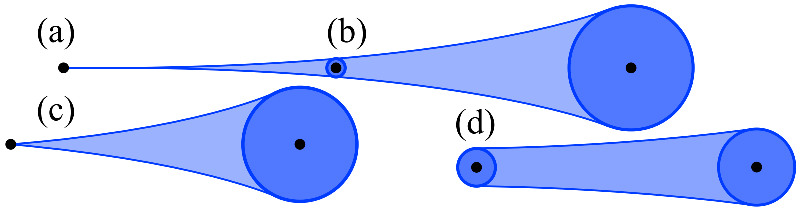

Figure 3. A graphical representation of how the positional and water current uncertainties evolve throughout an action where the black dots represent the expected positions and the blue shaded regions represent the uncertainty. The initial conditions are (a) the position and water current are both known, (b) the position and water current are both unknown, (c) the position is known and the water current is unknown, and (d) the position is unknown and the water current is known.

where $a \in \mathcal{A}$ consists of a non-measurement component a_M and a component measurement action. Note that the trajectory terminates when $c_{\text{p}}(x, y) \neq 0$.

With the water current and cost function formally defined, we can now define the movement of the submarine at any given time. For a specified movement vector M such that $\|M\|_2 \leq 1$, the movement of the submarine is given by $d(x, y, W, M, t) = (x, y) + t(M + W(d(x, y, M, t)))$. The non-measurement action corresponding to M is then defined as $a_M(x, y, W) = d(x, y, W, M, t')$, where $t' = \sup\{t \in [0, 1] | c_{\text{p}}(d(x, y, W, M, t)) < 100\}$. Note that $t' = 1$ only occurs if the agent does not crash into an island during the action.

For any chart, with or without water currents, we have $V_0 \equiv 0$ and $V_1(x, y) = \min_{a \in \mathcal{A}} c(x, y, a)$. Examples of a chart without and with water currents are shown in Figures 2(a) and 2(b) with V_1 for each case shown in Figures 2(c) and 2(d) respectively. If the water current is known, Bellman's equation for our system becomes

$$V_n(x, y) = \min_{a \in \mathcal{A}} c(x, y, a) + \gamma V_{n-1}(a_M(x, y, W)), \tag{4.1}$$

where $W \equiv 0$ for Figure 2(e) and $W = W(x, y)$ for Figure 2(f). In this system, (4.1) results in a converged value function V after finite n. The converged value functions for the examples above are shown in Figures 2(e) and 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 Figure 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 the value iteration algorithm for POMDPs in (2.3). 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 case** of four considered. As the water current changes when the submarine moves away from this position, uncertainty in the water current grows during any non-measurement action taken, leading to the growth of uncertainty in the position shown in Figure 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 chooses: 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 these estimates. This is the **second case**. As before, uncertainty in position increases due to uncertainty in the water current growing over time. However, uncertainty in position also increases due to the initial non-zero uncertainty in the water current. This combination leads to the growth of uncertainty in the position shown in Figure 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. The uncertainty growth rate is the rate the agent's uncertainty of the water current increases over each action. The agent's uncertainty of position increases relative to the uncertainty of the water current, not just the uncertainty growth rate. The maximum water current magnitude represents the true uncertainty that is present in the system, whereas the uncertainty growth rate represents the uncertainty the agent assumes is present in the system.

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 chooses to take a measurement. If the expected position of the submarine is inside the target region, the policy chooses to specifically take a GPS measurement. Otherwise, the policy chooses to 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 Figure 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 Figure 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 chooses to select the non-measurement action with the lowest expected value based on the new distribution of trajectories.

In this problem we are assuming the agent does not have access to all components of the POMDP tuple, therefore we must replace the hidden Markov model filter with something that incorporates the uncertainty of our system. This gives us the modified Q-function

$$\begin{aligned}\mathcal{Q}(\hat{x},\hat{y},\sigma_p,\hat{W},\sigma_w,a) = \frac{1}{16\sigma_p^2\sigma_w^2}\oiint_{\sigma_p}\oiint_{\sigma_w}&\Big(c(\hat{x}+x',\hat{y}+y',a)\\ &+\gamma V(a_M(\hat{x}+x',\hat{y}+y',\hat{W}+(w_x,w_y)))\Big)dw_x dw_y dx' dy',\end{aligned} \tag{4.2}$$

where $\oiint_\sigma$ is the 2D integration over the circular region of radius σ centered at the origin, $(\hat{x},\hat{y})$ is the agents estimate of their position, $\hat{W}$ is the agents estimate of the local water current, σ_w is the water current uncertainty, σ_p is the positional uncertainty. We also define

$$\begin{aligned}\mathcal{Q}(\hat{x},\hat{y},\sigma_p,\hat{W},0,a) &= \frac{1}{4\sigma_p^2}\oiint_{\sigma_p}\Big(c(\hat{x}+x',\hat{y}+y',a)+\gamma V(a_M(\hat{x}+x',\hat{y}+y',\hat{W}))\Big)dx'dy',\\ \mathcal{Q}(\hat{x},\hat{y},0,\hat{W},\sigma_w,a) &= \frac{1}{4\sigma_w^2}\oiint_{\sigma_w}\Big(c(\hat{x},\hat{y},a)+\gamma V(a_M(\hat{x},\hat{y},\hat{W}+(w_x,w_y)))\Big)dw_x dw_y,\\ \mathcal{Q}(\hat{x},\hat{y},0,\hat{W},0,a) &= c(\hat{x},\hat{y},a(\hat{x},\hat{y},\hat{W}))+\gamma V(a_M(\hat{x},\hat{y},\hat{W})).\end{aligned} \tag{4.3}$$

Our policy is then constructed by choosing the non-measurement action that minimizes the Q-function. If the expected cost of that action is greater than the cost of the measurement actions, then a measurement action is taken instead (with the GPS taking precedence over the current profiler).

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), state-space to a resolution of 152×152 for the value function, and integrals in (4.2). 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 agent's incomplete access to 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 Figure 1(b), inclusive of 0 and 1.

For the chart generation, we have $x_{\max} = y_{\max} = 10$ 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. 100 charts of low density density, 150 charts of medium density, and 250 charts of high density will be used with 10 different initial states each. The maximum water current magnitude and the linear rates at which the uncertainty used in a policy grows (uncertainty growth rate) will be parameters of experimentation, each varying from 0 to 1. The estimates in water currents are bounded by $\|\hat{W}(x, y)\|_2 \leq w_{\max}$. 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, it is possible that a trajectory can be cyclic and these (potentially) infinite trajectories are typically the only ones that lasted more than 25 steps. For this reason, we limit all trajectories to 25 steps. We consider the following three types of outcomes: a policy that reaches the target within 25 steps is considered successful, a policy that crashes within 25 steps is a failure, and a policy that neither is successful nor a failure.

For each chart, a value function is generated using the method described in Section 4.1. For each initial state, a policy is constructed several times using the method described in Section 4.2, where the uncertainty growth rate is varied. An uncertainty growth rate of zero is equivalent to using Bellman's equation and assuming there does not exist any water current.

Figures 4(a)-(c) show the success rate as a function of uncertainty growth rate and maximum water current for policies constructed for low, medium, and high island densities, respectively. When the maximum water current is zero, the uncertainty growth rate does not affect the agent's behavior due to the upper bound on the water current estimates. Without any water current, the problem is equivalent to the MDP problem initially used to generate the value functions. Hence, the agent succeeds in 100% of the charts for all three island density sets, which is expected as the agent has the true value functions for the problem, however, this is no longer true once the maximum water current is non-zero.

For an uncertainty growth rate of zero, the agent performs quite well at extremely low maximum water currents and lower island densities. However, even for low (non-zero) maximum water currents, the agent's performance begins to decline for charts with higher island densities, succeeding in less than 90% of charts. As the maximum water current increases the agent's performance steadily decreases to the point it succeeds in 0% of all charts. This drop to a 0% success rate is most notable in the charts with higher island densities.

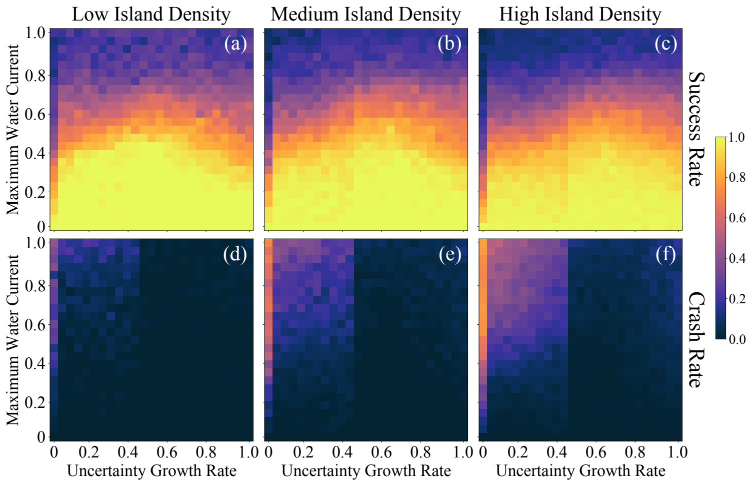

Figure 4. Policy statistics constructed over 500 unique charts for various uncertainty growth rates and maximum water currents. (a)-(c) The agent's success rate, where success means the agent navigated the submarine into the target area for low, medium, and high island densities, respectively. (d)-(f) The agent's crash rate for low, medium, and high island densities respectively. Note that the agent's success and crash rates do not include the cases where the agent's trajectory lasts more than 20 steps. In all cases, a maximum water current of 0 is equivalent to no water current existing and an uncertainty growth rate of zero is equivalent to using standard DP for MDPs.

Excluding a few outliers for the more extreme maximum water currents, even the smallest tested non-zero uncertainty growth rate outperforms the zero case in charts of all island densities. For maximum water currents less than 0.45, 0.4, and 0.35, there exists at least one tested uncertainty growth rate that gives the agent a success rate of 100% for all charts of low, medium, and high island densities, respectively. At those maximum water currents, when using a non-zero uncertainty growth rate the agent is able to get an increase in success rate of up to 58%, 67%, and 63% for all charts of low, medium, and high island densities.

While the specific non-zero value for the uncertainty growth rate does not make much difference in the agent's success rates at lower maximum water currents, it matters significantly for larger maxima. For the larger maximum water currents, the agent's success rate increases on average as the uncertainty growth rate increases (approximately 0.7). The agent's success rate begins to decline on average once the uncertainty growth rate is increased beyond this point. At these high uncertainty growth rates, any target remotely close to an island appears too risky to reach, i.e. the expected cost due to crashing is greater than the expected negative cost of succeeding.

The trends discussed here all tend to break for the largest maximum water currents as the agent's success rate stays close to 0%. For maximum water currents near 1.0, the displacement caused by the water current can be as large as the distance the agent can possibly cover in a single action. This can make it impossible for the agent to overcome the water current and reach the target in most cases, regardless of the uncertainty growth rate or method used.

Figures 4(d)-(f) instead show the crash rate as a function of uncertainty growth rate and maximum water current for policies constructed for low, medium, and high island densities, respectively. In the cases the agent's success rate is near 100% the crash rate must be near 0%, however lower success rates do not imply higher crash rates. For an uncertainty growth rate of zero, the agent's crash rate increases at a similar rate to the decrease in success rate as the maximum water current is increased, reaching 80% in some cases.

When the maximum water current is increased, we see a similar trend for the lower non-zero uncertainty rates as before, where the crash rate increases simultaneously as the success rate decreases. The increase in crash rate is much less significant compared to the zero case though, even as the success rate goes to 0%. Therefore, even when the agent does not succeed, it is much better at managing to avoid islands. For the largest of the maximum water currents, even the smallest non-zero uncertainty growth rate decreases the crash rate by over 25%.

Regardless of the success rate, we see the crash rate drop to (or near) 0% for larger uncertainty growth rates. The larger uncertainty growth rates use the extremes of water current estimates, therefore the agent uses most actions avoiding islands, rather than reaching the target. The crash rate slightly increases again for the charts with higher island densities at larger uncertainty growth rates and maximum water currents. In these cases, the uncertainty of each action is so large that the agent estimates they will all end in crashing. The "safest" action in these cases is then to do nothing, in which case the water current causes the agent to drift into an island, resulting in a crash.

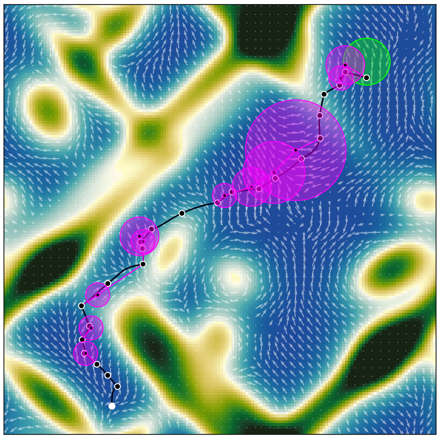

Figure 5. An example trajectory of a successful policy on a high island density chart. The white circle represents the agent's starting position. The black lines represent the agent's true trajectory, where each white outlined circle indicates the points when the agent was required to select an action. The magenta lines represent the agent's estimated trajectory, where the transparent magenta shading represents the magnitude of uncertainty and each magenta outlined circle represents when the agent was required to select an action. When a white-outlined circle is connected to a magenta one, this represents when the agent takes an action and does not use the GPS measurement. When a magenta-outlined circle is connected to a white one, or when no magenta-outlined circle is present for that step, this represents when the agent takes an action and then uses the GPS measurement. The transparent green circle represents the target region.

In every case, the agent uses slightly less than one measurement per action on average. If the agent were to use both the GPS and current profiler measurements for every action, it would result in an average measurement cost of 0.55 per action. Our agent's measurement costs fall into three regimes based on the uncertainty growth rate: approximately 0.2, 0.3, and 0.45 for uncertainty growth rates less than 0.25, between 0.25 and 0.45, and greater than 0.45, respectively. For small uncertainty growth rates, we have a large reduction, and for large uncertainty growth rates we have a minor, but non-zero, reduction in measurement costs. This tells us to choose the smallest uncertainty growth rate that results in the largest success rate for any given maximum water current, thus minimizing measurement costs without sacrificing navigational safety. An example of a successful policy trajectory in a high island density chart with a non-zero water current magnitude is shown in Figure 5.

7. Conclusions & Future Work

Motivated by the real-world applicability of POMDPs and systems with uncertainty, we have shown that incomplete access to information can be leveraged with DP methods to construct navigational policies that both maintain safety and reduce total measurement cost. The 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 (uncertainty growth rate of zero), the policies produced using these value functions perform very poorly. However, when uncertainty methods are included, the success rate on average is doubled and the crash rate is brought to (or nearly to) zero.

While the method shown here has been quite successful, it is not perfect. The success of using a fixed uncertainty growth rate makes the assumption that the maximum water current is known. We would like to include an adaptive uncertainty growth rate in future versions of this algorithm. This adaptive method could be a neural network-based learned mapping from charts to optimal uncertainty growth rate for that trajectory or a constantly updating value based on calculated average water currents between GPS measurements. For further comparison of our method's performance, we would like to develop a deep RL-based policy as well.

Acknowledgements

C. Beeler and C. Bellinger performed work at the National Research Council of Canada under the AI4D program. I. Tamblyn acknowledges NSERC. Additional information and 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". In: *Bulletin of the American Mathematical Society* 60.6 (1954), pp. 503–515.

[3] C. Boutilier, R. Dearden, and M. Goldszmidt. "Stochastic dynamic programming with factored representations". In: *Artificial intelligence* 121.1-2 (2000), pp. 49–107.

[4] Z. Zamani, S. Sanner, and C. Fang. "Symbolic dynamic programming for continuous state and action mdps". In: *Proceedings of the AAAI Conference on Artificial Intelligence.* Vol. 26. 2012.

[5] B. Bakker, Z. Zivkovic, and B. Krose. "Hierarchical dynamic programming for robot path planning". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE. 2005, pp. 2756–2761.

[6] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction.* 2018.

[7] R. S. Sutton, D. Precup, and S. Singh. "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning". In: *Artificial intelligence* 112.1-2 (1999), pp. 181–211.

[8] A. L. Strehl, L. Li, and M. L. Littman. "Reinforcement Learning in Finite MDPs: PAC Analysis." In: *Journal of Machine Learning Research* 10.11 (2009).

[9] T. Wu, Y. Yang, S. Du, and L. Wang. "On Reinforcement Learning with Adversarial Corruption and Its Application to Block MDP". In: *International Conference on Machine Learning.* PMLR. 2021, pp. 11296–11306.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (2013).

[11] 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". In: *nature* 518.7540 (2015), pp. 529–533.

[12] E. van der Pol, D. Worrall, H. van Hoof, F. Oliehoek, and M. Welling. "MDP homomorphic networks: Group symmetries in reinforcement learning". In: *Advances in Neural Information Processing Systems* 33 (2020).

[13] 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.

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

[15] S. Seuken and S. Zilberstein. "Memory-Bounded Dynamic Programming for DEC-POMDPs." In: *IJCAI.* 2007, pp. 2009–2015.

[16] K. G. Papakonstantinou and M. Shinozuka. "Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation". In: *Reliability Engineering & System Safety* 130 (2014), pp. 214–224.

[17] S. Sanner and K. Kersting. "Symbolic dynamic programming for first-order POMDPs". In: *Twenty-Fourth AAAI Conference on Artificial Intelligence.* 2010.

[18] E. A. Hansen, D. S. Bernstein, and S. Zilberstein. "Dynamic programming for partially observable stochastic games". In: *AAAI.* Vol. 4. 2004, pp. 709–715.

[19] D. Lee, N. He, and J. Hu. "Dynamic programming for POMDP with jointly discrete and continuous state-spaces". In: *2019 American Control Conference (ACC).* IEEE. 2019, pp. 1250–1255.

[20] 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.* PMLR. 2021, pp. 9744–9755.

[21] K. Azizzadenesheli, A. Lazaric, and A. Anandkumar. "Reinforcement learning of POMDPs using spectral methods". In: *Conference on Learning Theory.* PMLR. 2016, pp. 193–256.

[22] S. Bhattacharya, S. Badyal, T. Wheeler, S. Gil, and D. Bertsekas. "Reinforcement learning for POMDP: Partitioned rollout and policy iteration with application to autonomous sequential repair problems". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 3967–3974.

[23] D. Steckelmacher, D. M. Roijers, A. Harutyunyan, P. Vrancx, H. Plisnier, and A. Nowé. "Reinforcement learning in POMDPs with memoryless options and option-observation initiation sets". In: *Thirty-second AAAI conference on artificial intelligence.* 2018.

[24] V. Krishnamurthy. "Convex Stochastic Dominance in Bayesian Localization, Filtering, and Controlled Sensing POMDPs". In: *IEEE Transactions on Information Theory* 66.5 (2019), pp. 3187–3201.

[25] E. Akyol, U. Mitra, and A. Nayyar. "Controlled sensing and event based communication for remote estimation". In: *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton).* IEEE. 2014, pp. 545–549.

[26] D.-S. Zois and U. Mitra. "Controlled sensing: a myopic fisher information sensor selection algorithm". In: *2014 IEEE Global Communications Conference.* IEEE. 2014, pp. 3401–3406.

[27] C. Bellinger, R. Coles, M. Crowley, and I. Tamblyn. "Active Measure Reinforcement Learning for Observation Cost Minimization". In: *Canadian Conference on Artificial Intelligence.* Lecture Notes in Computer Science (LNCS). Springer. Springer, 2021, p. 12.

[28] H. Nam, S. Fleming, and E. Brunskill. "Reinforcement Learning with State Observation Costs in Action-Contingent Noiselessly Observable Markov Decision Processes". In: *Advances in Neural Information Processing Systems* 34 (2021).

[29] 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". In: *Health affairs* 28.5 (2009), pp. 1475–1484.

[30] C. Beeler, S. G. Subramanian, K. Sprague, N. Chatti, C. Bellinger, M. Shahen, N. Paquin, M. Baula, A. Dawit, Z. Yang, et al. "ChemGymRL: An Interactive Framework for Reinforcement Learning for Digital Chemistry". In: *arXiv preprint arXiv:2305.14177* (2023).

[31] R Zaccone, E Ottaviani, M Figari, and M Altosole. "Ship voyage optimization for safe and energy-efficient navigation: A dynamic programming approach". In: *Ocean engineering* 153 (2018), pp. 215–224.

[32] 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". In: *Sensors* 19.2 (2019), p. 434.

[33] A. Fan, Z. Wang, L. Yang, J. Wang, and N. Vladimir. "Multi-stage decision-making method for ship speed optimisation considering inland navigational environment". In: *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 235.2 (2021), pp. 372–382.