Information-Lookahead Planning for AUV Mapping

Technical Report CSR-09-01*
School of Computer Science, University of Birmingham

Zeyn A Saigol
Richard W Dearden
Jeremy L Wyatt
School of Computer Science
University of Birmingham, UK
{zas, rwd, jlw}@cs.bham.ac.uk

Bramley J Murton
National Oceanography Centre
Southampton, UK
bjm@noc.soton.ac.uk

Abstract

Exploration for robotic mapping is typically handled using greedy entropy reduction. Here we show how to apply information lookahead planning to a challenging instance of this problem in which an Autonomous Underwater Vehicle (AUV) maps hydrothermal vents. Given a simulation of vent behaviour we derive an observation function to turn the planning for mapping problem into a POMDP. We test a variety of information state MDP algorithms against greedy, systematic and reactive search strategies. We show that directly rewarding the AUV for visiting vents induces effective mapping strategies. We evaluate the algorithms in simulation and show that our information lookahead method outperforms the others.

1 Introduction

How should a robot that is creating a map from data explore its environment to build the most effective map as quickly as possible? We explore this problem as one of information state space planning when using Autonomous Underwater Vehicles (AUVs) for underwater mapping. We address the specific problem of prospecting for hydrothermal vents, which are superheated outgassings of water found on mid-ocean ridges, and are of great interest to marine scientists.

Hydrothermal vents are formed where continental plates diverge and can be detected from a distance because they emit a chemical-containing plume that rises several hundred metres above the vent, and is spread by diffusion and ocean currents over an area dozens of kilometres wide. The plume is detected using a passive sensor that updates as the AUV moves. Detecting a plume gives only limited information about the location of the source vent, for several reasons; first, because turbulent flows and changing currents mean there is no steady chemical gradient that can be traced to the vent. Second, it is not possible to relate the concentration of chemicals in the plume directly to the distance from a vent, because some vents are more active than others. Finally, vents are often found in clusters (known as vent fields), and plumes from several vents will mix. Current methods for mapping hydrothermal vents employ a series of nested box surveys in which the AUV "mows the lawn" in progressively smaller areas, chosen after analysis of each survey. This is inefficient. If the AUV could perform on-board data analysis and planning, rather than following an exhaustive search pattern, it could cover a much larger area while still mapping all or most vents. The domain is described in more detail in [Dearden *et al.*, 2007].

*This report is an extended version of a paper appearing in IJCAI'09

Our problem decomposes into two parts, mapping and planning. In mapping the AUV infers a map of likely locations for vents based on the observation history. As stated earlier the vent mapping problem is challenging because the AUV can refine its map for a particular location using information from plume detections elsewhere. This is different from mobile robot mapping using laser or vision since there the robot must be in the vicinity of a location to draw sensory information about it. We focus on planning, and so we adopt the recent vent mapping algorithm of Jakuba [2007]. This gives the AUV an occupancy grid (OG) map, which gives the probability that there is a vent for each location.

The planning problem is as follows: Given such an OG map, choose where to go in order to localise the vents as efficiently as possible. Again because of the distal clues provided by plume detections this planning for mapping problem is different from that tackled to support conventional mobile robot mapping. For planning purposes the actions of the AUV are simply moving in the world. We assume that sensing occurs at every step, returning either that a vent has been found, that the plume from a vent has been detected, or nothing. The challenge is that we don't just want to visit potential vents given our current probability distribution over their locations, but also to choose actions that give us more information about where the vents might be. This kind of information-gathering problem is most easily modelled as a partially observable Markov decision problem (POMDP), but here the state space is far too large to be solved by conventional POMDP methods.

The approach we take is to use our current state to constrain the reachable state space and to do forward search in the belief space to identify reachable states and evaluate them. Unfortunately, even with deterministic actions, the branching factor of this search tree quickly makes it impractical to evaluate, and we consider a number of possible heuristics to evaluate the leaves of the tree. Details of the algorithm are given in Section 3. Despite the limited lookahead that is computationally feasible, the results we obtain show this to be an effective approach in practice. However, the most intuitively plausible heuristic we apply turns out to have surprisingly poor performance in practice (see Section 5 and the discussion of why this occurs in Section 6). We also compare the performance of our algorithm to two commonly-used approaches, a pre-programmed mow-the-lawn search pattern and a reactive approach similar to chemotaxis [Farrell *et al.*, 2003], neither of which are able to find vents as effectively as the limited-lookahead approach we describe.

2 Problem Definition

2.1 Mapping Algorithm

As we said above, we rely on an existing mapping algorithm to compute an OG map based on observations [Jakuba, 2007]. It is designed specifically for creating a map of hydrothermal vents using plume detection data obtained by an AUV. The OG approach [Martin and Moravec, 1996, Elfes, 1989] works by dividing the space into a regular grid, and calculating cell occupancy based on a series of sensor readings. In the AUV domain, being occupied means containing a vent; however classic OG algorithms do not work for vent mapping because they assume that a sensor reading is determined by a single test cell, and is independent of previous sensor readings. In contrast, for vent mapping the sensor can register a plume emitted from any of a large number of neighbouring cells, and previous sensor readings are vital to help narrow down which cells are likely to have been responsible. As Jakuba points out, the errors induced by this classic OG assumption are exacerbated when the prior probability of occupancy is very low (as it will be for vents, because only a handful of vents will be found over a search area of several square kilometres).

Jakuba develops a more accurate OG approach, firstly by compressing readings from multiple sensors into a binary detection or non-detection of a hydrothermal plume. Given this binary detection input, he is able to develop a tractable approximate algorithm that produces significantly better results in low-prior environments than the standard OG algorithm. The approach relies on a forward model of the behaviour of the plume produced by a vent, which is used to estimate, for each OG cell, the likelihood that a vent

in that cell would generate a detection at the location of the vehicle.

The OG mapping approach implies we have to divide the search area into a grid of C cells, each of which can either contain a vent or not. Since the map is two-dimensional, we restrict the agent to move in two dimensions, which also simplifies the planning problem. We also require the agent to move only between adjacent grid cells (i.e. only North, East, South or West). Actions are assumed to be deterministic, as AUV ground-tracking navigation is quite reliable. Similarly, we assume that the agent can move one cell in one timestep, and there are a fixed number of timesteps in a mission, which ignores the fact that certain paths and environmental conditions may drain the battery faster than others.

2.2 POMDP Formulation

This problem is best formally represented as a partially observable Markov decision process (POMDP), because the system state includes the location(s) of nearby vents, which cannot be directly observed by the agent. A POMDP is defined by the tuple $\langle S, A, T, \mathcal{R}, Z, H \rangle$, where S is a set of possible states, A is a set of actions available to the agent, $T = P(s'|s, a)$ is a state transition function, $r = \mathcal{R}(s, a)$ is a reward function, Z is a set of possible observations, and finally $H = P(z|s, a)$ is an observation function. The aim is for the agent to maximise its expected future discounted reward, where the total reward following time t is given by $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$, using a discount factor γ, $0 < \gamma < 1$. We now present the exact model of the problem we use, based on the requirements of planning within an OG framework.

2.2.1 State Space

The state s is composed of c_{AUV}, the cell the AUV is in; $\mathbf{U}$, the ocean current vector; $\mathbf{m}$, the actual vent locations (a length-C vector of booleans indicating whether each cell contains a vent); and $\mathbf{v}$, a similar vector marking all vents that have already been found. If $\mathbf{U}$ is discretised into $d \times d$ bins, there are $C \cdot d^2 \cdot 2^C \cdot 2^C$ possible states.

2.2.2 Actions and Transition Function

The actions a are to move N/E/S/W, and the transition function $P(s'|s, a)$ is deterministic. Actions that would leave the search area are not allowed.

2.2.3 Observations and Observation Function

Although the input to the OG algorithm is a single observation p (whether or not a hydrothermal plume is detected), our agent also needs to know when it has found a vent, to ensure it doesn't try to claim rewards by visiting the same vent multiple times. We therefore introduce a deterministic vent detector l, so the possible observations when moving into cell c are:

$$l = \begin{cases} 1 & \text{if a vent is found in cell } c \\ 0 & \text{otherwise} \end{cases}$$

$$p = \begin{cases} 1 & \text{if a plume is detected in cell } c \\ 0 & \text{otherwise} \end{cases}$$

The observation function $P(z|s, a)$, where $z = (l, p)$, can be found from Jakuba's forward model.

2.2.4 Reward Function

$$\mathcal{R}(s, a, s') = \begin{cases} R_{vent} & \text{if } s' \text{ is an } \textit{unvisited} \text{ vent} \\ 0 & \text{otherwise} \end{cases}$$

where R_{vent} is a positive constant.

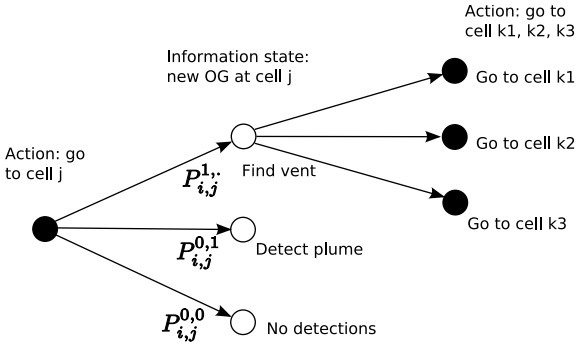


Figure 1: Information-State MDP, where black circles represent actions and white circles are belief states.

2.3 ISMDP Formulation

To find a policy in the POMDP defined above we use the standard approach for POMDP solving and translate it to an *information state MDP* (ISMDP), where the state space is the probability distribution over the possible states of the POMDP, known as the *belief state*. While the belief state is fully observable by the agent, it is continuous and therefore standard MDP techniques cannot be applied. The components $\langle S, A, T, \mathcal{R} \rangle$ of the ISMDP are described below.

2.3.1 Belief State Space

In our ISMDP, the belief state b is comprised of:

- c_{AUV}, the cell the AUV is in; $\mathbf{U}$, the current; and $\mathbf{v}$, the list of previously-found vents (all of which are assumed to be known exactly).
- The occupancy grid $\mathbf{O}$. $\mathbf{O}$ is a length-C vector of values $P(m_c)$, the probability of cell c containing a vent (which will be either zero or one for visited cells). The OG defines a distribution over true vent locations $\mathbf{m}$.

Note that using an OG approach has considerably reduced our belief state space (the map contributes only C variables to the belief state instead of 2^C) because the occupancy grid is an approximation where we assume the probability of each cell being occupied is independent.

2.3.2 Actions

The actions in the ISMDP are identical to those in the POMDP, deterministically moving N/E/S/W.

2.3.3 Transition Function

The transition function is given by

$$P(b'|b,a) = \sum_{z \in Z} P(b'|b,a,z) P(z|b,a)$$

where $P(b'|b,a,z)$ is zero for all belief states except the one where b' is the output from the OG algorithm applied to b, a, z, and the observation model $P(z|b,a_c)$ is the probability of observing $z = (l,p)$ in cell c. Section 2.4 gives our derivation of this observation model, based on the same plume model Jakuba uses for his OG algorithm. Figure 1 shows the action/observation transitions for the ISMDP.

2.3.4 Reward Function

The reward function for an action moving into cell c is $\rho(b, a_c) = \sum_{s \in S} b(s)\mathcal{R}(s, a_c)$, but $\mathcal{R}(s, a_c) = 0$ for all states except ones where the destination cell c contains a vent. As the occupancy grid values for each cell represent the probability of it containing a vent independent of the rest of the grid, the reward function is $\rho(b, a_c) = P(m_c)R_{vent}$ for all cells c that have not been visited, and zero for those that have, because either there is definitely not a vent in them, or we have already found that vent.

2.4 Derivation of Observation Function

We derive the observation model $P(z|b, a)$ from the forward model of [Jakuba, 2007]. We write $P_{ij}^{l,p}(\mathbf{O})$ for the probability of observing l, p when moving from cell i to cell j given OG $\mathbf{O}$, and O_i for the probability according to $\mathbf{O}$ of a vent in cell i. Further we denote cell c containing a vent by m_c (so $O_c = P(m_c)$), and cell c being empty by $\tilde{m}_c$.

2.4.1 Probability of Locating a Vent

If the AUV is in the same cell as a vent then it is overwhelmingly likely to detect a plume from that vent, so we ignore plume detections when we find a vent. Given we assume the vent detector is deterministic, then by definition:

$$P_{ij}^{1,\cdot}(\mathbf{O}) = O_j$$

2.4.2 Plume Model

To find the probability of detecting a plume but no vent in cell j, we need to use the forward model for a plume, the existing OG $\mathbf{O}$, and the current, $\mathbf{U}$.

Jakuba's plume model assumes particles emitted from source location $\mathbf{x}_s$ at time $t - \tau$ are Gaussian-distributed at time t:

$$\mathbf{x}_p \sim N(\mathbf{x}_s + \tau\mathbf{U}, \tau\sigma^2)$$

If the current is variable rather than fixed, $\tau\mathbf{U}$ must be replaced by $\sum_{r=t-\tau}^{t-1} \mathbf{U}_r$

2.4.3 Probability of Plume Detection From a Single Cell

Given a vehicle location $\mathbf{x}_v$ and source vent location $\mathbf{x}_s$, we want to find the probability of observing a given particle p. Following Jakuba §5.2.1, we assume a particle has a radius b, and will always be detected if $|\mathbf{x}_p - \mathbf{x}_v| < b$. Then Jakuba uses the approximation

$$\begin{aligned} P\left(|\mathbf{x}_p - \mathbf{x}_v| < b\right) &\approx b^2 P\left(\mathbf{x}_p = \mathbf{x}_v\right) \\ &= b^2 \frac{1}{\sqrt{2\pi\tau\sigma^2}} \exp\left(-\frac{|\mathbf{x}_s + \tau\mathbf{U} - \mathbf{x}_v|^2}{2\tau\sigma^2}\right) \end{aligned}$$

Next we need to consider all the particles emitted from a given source, i.e. we want

$$P(d_c|m_c) = P\left(\text{detect}(p_c^1) \cup \text{detect}(p_c^2) \cup \ldots \cup \text{detect}(p_c^n)\right)$$

where d_c represents a plume detection due to a source at c, the source $\mathbf{x}_s$ is assumed to lie in the cell m_c, and p_c^m represents the m^{th} particle emitted from a source at c. Assuming that the detection of each particle is independent, and using $P(d_c|m_c) = 1 - P(\neg d_c|m_c)$, we get

$$P(d_c|m_c) = 1 - \prod_n \left(1 - P(\text{detect}(p_c^n)\right)$$

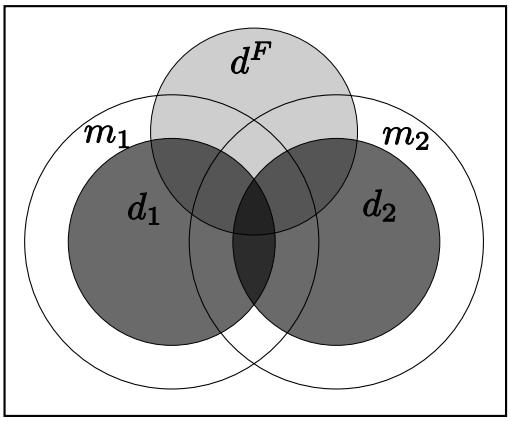


Figure 2: Venn diagram of the occupancy of cells m_c, plume detections due to specific cells d_c, and a false positive plume detection d^F

Note that $P(d_c|m_c)$ is identical to the quantity P_s^t in [Jakuba, 2007], and the equation above is essentially the same as equation (5.9) in Jakuba §5.2.1, although he derives this equation differently.

In the simple environmental simulation we use (which is based on Jakuba's implementation), each source emits one particle at each timestep, and we further assume that only the particle with the nearest expected location contributes to the probability of a detection, i.e. $\sigma \ll |\mathbf{U}|$.

2.4.4 Probability of Plume Detection with Full OG Map

To extend this to a full map, where multiple cells may be occupied, we need the union of the probability of a detection caused by each cell, but where these probabilities are "weighted" by the probability of that cell being occupied.

We know $P(d_c|m_c)$ and $P(m_c)$, and we are interested in $P(d)$, the probability of a detection from any cell. It can be seen from the Venn diagram in Figure 2 that

$$P(d) = P\left(d^F \cup d_1 \cup d_2 \cup \ldots \cup (d_C)\right)$$

where d^F is a false positive detection. It is also clear from the Venn diagram that $P(d_c) = P(d_c \cap m_c) = P(d_c|m_c)P(m_c)$.

If we assume all the events $(d_c \cap m_c)$ are independent, we get

$$P(d) = 1 - \left(1 - P^F\right) \prod_{c=1}^{C} \left(1 - P(d_c|m_c)P(m_c)\right)$$

where P^F is the probability of a false positive plume detection.

However, to find $P_{ij}^{0,1}$ we actually need $P(d \cap \tilde{m}_j)$, as if m_j is true then a vent will be detected and the branch described in section 2.4.1 will be followed. We find

$$P(d \cap \tilde{m}_j) = P(d|\tilde{m}_j)P(\tilde{m}_j) = \left(1 - P(\neg d|\tilde{m}_j)\right)\left(1 - O_j\right)$$

and

$$\begin{aligned} P(\neg d|\tilde{m}_j) &= \frac{P(\neg d \cap \tilde{m}_j)}{P(\tilde{m}_j)} \\ P(\neg d \cap \tilde{m}_j) &= P(\neg d^F \cap \neg d_1 \cap \neg d_2 \cap \ldots \cap \neg d_C \cap \tilde{m}_j) \\ &= \left(1 - P^F\right) P(\tilde{m}_j) \prod_{c \neq j} (1 - P(d_c)) \end{aligned}$$

as $P(\neg d_j|\tilde{m}_j) = 1$ (if there is no vent at cell j, we can't get a detection due to that cell). So

$$P_{ij}^{0,1}(\mathbf{O}) = P(d \cap \tilde{m}_j) = (1 - O_j) \left(1 - \left(1 - P^F\right) \prod_{c \neq j} (1 - P(d_c))\right)$$

where $P(d_c) = P(d_c|m_c)P(m_c)$.

2.4.5 Probability of Detecting Neither Plume Nor Vent

The probability of this event is $P(\neg d \cap \tilde{m}_j)$, which appears in Section 2.4.4 above.

$$\begin{aligned} P_{ij}^{0,0}(\mathbf{O}) &= P(\neg d \cap \tilde{m}_j) \\ &= P(\tilde{m}_j)P(\neg d|\tilde{m}_j) \\ &= (1 - O_j)\left(1 - P^F\right) \prod_{c \neq j} (1 - P(d_c)) \end{aligned}$$

3 Algorithm

State-of-the-art POMDP solvers are able to solve some classes of problems with as many as 30 million states [Poupart and Boutilier, 2004]. With a grid size of 100×100, which is adequate for a real ocean survey [Jakuba and Yoerger, 2008], our problem has over 4^{10000} states, so even with the factored belief space due to using an OG it is far too large to be solved by existing algorithms. We find an approximate policy by forward search from the current belief state.

3.1 Information Lookahead

The *information lookahead* (IL) algorithm is an approximate method for solving an ISMDP by only evaluating the value of actions up to N steps into the future. The value of actions can be calculated in a recursive way: for each action, find the new belief states we may end up in, and then for each new belief state, find the value of each possible action. The limited lookahead makes Q, the value of an action a given a belief state b, also a function of r, the number of steps left to take—the same belief state may be worth more if the agent can take more steps following it. r is initially set to N, and is reduced by one on each recursive evaluation. When the recursion ends, at $r = 0$, we have to provide a $Q(b, a)$ value, which could be an arbitrary value such as zero or be heuristically chosen. For $r > 0$ the Q-value is given by:

$$Q(b, a, r) = \rho(b, a) + \gamma \sum_z P(z|b, a) \max_{a'} Q(b', a', r - 1)$$

where γ is the discount factor, $\rho(b, a)$ and $P(z|b, a)$ are the reward and observation functions defined above and b' is calculated using Jakuba's OG update algorithm.

After looking ahead N steps, a heuristic value has to be chosen for the terminal belief states, which should approximate the expected future reward following on from that belief state. A basic heuristic is

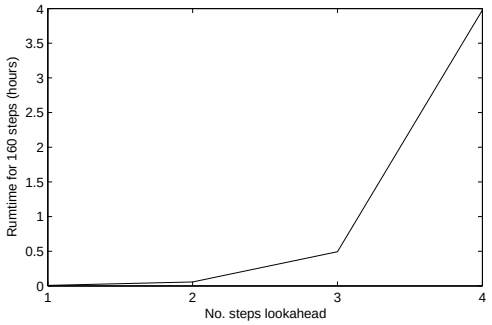


Figure 3: Run-time of the IL-CE algorithm for different values of N, the number of steps of lookahead

```
loop
   loop s ∈ S
      loop a ∈ A
         Q(s,a) = R(s,a) + γV(s')
      end
      V(s) = max_a Q(s,a)
   end
until ΔV < ε
```

Figure 4: CE value iteration algorithm, based on standard dynamic programming. ϵ specifies the convergence tolerance.

just to take the reward gained at the agent's final location, $\rho(b, a)$, as the value, and results using this heuristic are presented in Section 5, under the label of 'IL algorithm'. However, this algorithm takes no account of OG values more than $N + 1$ steps away from the agent's starting cell, so it was not expected to perform well.

As can be seen from Figure 1, the algorithm has a branching factor of nine: for each extra step taken, nine times as many Q-values must be evaluated. This exponential increase in calculations limits the lookahead N that can be practically used; most experiments have been performed with $N = 4$. Figure 3 shows the algorithm run-time on a 20×20 grid for different values of N. Therefore, the algorithm is quite dependent on a good heuristic to find values of belief states in the base case, and a more sophisticated heuristic is described in Section 3.2.

3.2 Certainty Equivalent Heuristic

A more sophisticated heuristic is the *certainty equivalent* (CE) heuristic, which finds a value for each cell in the grid by assuming that the agent's current belief state is correct. In other words, it assumes that future observations will not improve the agent's estimate of vent locations. This collapses the problem to an MDP where the state s is just the cell the agent is in (dropping the list of visited vents and fixing the OG), and this MDP can be solved by value iteration much faster than the IL version of the problem.

The CE algorithm is shown in Figure 4, and is slightly simpler than the standard dynamic programming algorithms (where the Q-value update is given by $Q(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a)V(s')$), as we have a deterministic transition function so do not have to sum over future states. Also, note that we are solving the infinite-horizon version of the MDP, in other words we are assuming that the agent can take an infinite number of steps and relying on the discount factor $\gamma < 1$ to produce finite Q-values.

The CE heuristic is not optimal because it takes no account of potential information gain, but also because it allows the agent to be rewarded multiple times from visiting the same cell. For the IL algorithm,

keeping track of the path of the agent doesn't result in any extra computational complexity, because it evaluates paths explicitly; however, adding an (arbitrary length) list of previously visited cells to the state space of the MDP would make the problem nearly as intractable as the original POMDP.

4 Previous Work

The overall vent finding problem is quite similar to SLAM [Thrun *et al.*, 2005]. The key difference here is that observations give information about a large number of cells in the occupancy grid (due to ocean current acting on the water from the vent), whereas in SLAM the number of cells that can affect a particular observation is very small. This affects the mapping in that it makes it very hard to determine which cell caused a particular observation. It also affects the planning because it means that the information that can be gained is not uniform over the grid—cells at the down-current end of the grid allow observations of more potential vent locations than those at the up-current end. This makes the planning problem significantly harder than for most SLAM applications, where greedy entropy reduction techniques are popular [Thrun *et al.*, 2005], although there has been recent work on lookahead methods [Kollar and Roy, 2008, Martinez-Cantin *et al.*, 2007].

The problem we address is closely related to the challenge set in [Bresina *et al.*, 2002], namely an oversubscribed planning problem with uncertainty and continuous quantities. In our case we do not consider the continuous quantities, although in a more realistic model we would have to consider the battery level of the vehicle, and we treat the actions as deterministic, so in this sense our problem is easier. However, it has one very significant difference in that the Bresina paper assumes that the problem is completely observable. In our case, although we know what location the AUV is in at every time step, we don't know where the vents are. This turns the problem from an MDP with a continuous state space to a POMDP. This means that techniques such as those in [Hauskrecht and Kveton, 2004, Feng *et al.*, 2004, Mausam *et al.*, 2005], although they can solve fairly large problems, aren't directly applicable to this problem.

The IL algorithm is a variation of the online POMDP algorithms discussed in [Ross *et al.*, 2008]. These algorithms use forward search together with a heuristic to evaluate leaf nodes, where most heuristics rely on solving a simplified version of the POMDP offline (for example, Q_{MDP} [Littman *et al.*, 1995]) where future observations are ignored). As these simplified versions have the same underlying state space as the full POMDP, they are not applicable in our domain where the number of states precludes any approach that assigns a value to every possible state. Paquet *et al.* [2005] introduce the RTBSS algorithm and apply it to a domain with a similarly large state space, but they don't describe the heuristic they use.

One can think of this as an observation planning problem, in the sense that we are choosing observations to make in order to maximise the number of vents found. Related work in this area includes [Darrell, 1997], which uses reinforcement learning in POMDPs to learn where to look in a gesture recognition system, and [Sridharan *et al.*, 2008], which plans which object-recognition algorithms to apply to an image. While these are both solving the problem of what observations to make, both are much more constrained than our problem, in particular because each observation cost is independent, unlike our case where moving to a location to make an observation is a significant part of the cost.

5 Experiments and Results

We compared our algorithms to two existing approaches: mowing-the-lawn (MTL) and chemotaxis. Mowing-the-lawn means moving in straight, parallel tracklines, with a short perpendicular movement at either end of the line to get to the next trackline. This approach is commonly used in marine exploration to survey an area, and the trackline spacing can be varied to obtain different survey resolutions.

The second comparison planner was a chemotaxis method loosely based on the behaviour of the male silkworm moth, as described in [Russell *et al.*, 2003]: on detecting a plume it firstly surges directly

up-current for six timesteps, and then searches in an increasing-radius spiral. If it detects a plume again at any point, the behaviour is re-started from the beginning. Prior to the first detection, our chemotaxis algorithm follows an MTL pattern, and if it tries to exit the search area at any point, it is redirected on a linear path approximately toward the centre of the area at a randomly-chosen angle.

We evaluated four different versions of the MDP-like algorithms: the IL-CE algorithm with a lookahead of four steps, the IL algorithm (using the basic heuristic of immediate reward) with lookaheads of both $N = 1$ and $N = 4$, and a CE algorithm where no lookahead was performed. All algorithms with discounting were run with $\gamma = 0.9$. Experiments were performed on a 20×20 grid and run for 160 timesteps. We used five different experimental configurations where we fixed the agent's start location and the position and number of vents. This allows a better comparison between algorithms under different conditions, as starting down-current provides a significant advantage over starting up-current of the vent, as the agent is much more likely to detect the plume coming from the vent. The configurations were:

v1down A single vent, with the agent starting down-current of the vent.

v1up Single vent, starting up-current of the vent.

v2down Two vents, starting down-current of both vents.

v2up Two vents, starting up-current of both vents.

v5 Five vents, starting halfway between the up- and down-current ends of the grid.

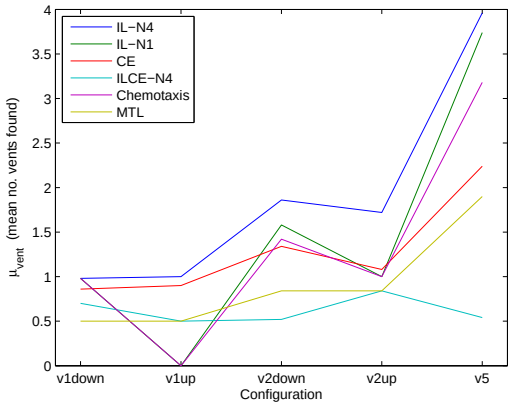
All configurations were run for 50 trials with each algorithm, where each trial used a different random seed. For the MDP algorithms, the only source of randomness was the dispersion of plumes from the vents; for chemotaxis, the return-to-centre angle was also chosen randomly; however, the MTL planner was independent of the plume, so to get a measure of the expected performance of MTL we fixed the agent's start location, but chose random locations for the vents.

We use two main criteria for evaluating the performance of vent prospecting algorithms: μ_{vent}, the number of vents found, and R, the return or total reward (both averaged over all trials). The return is given by $R = \sum_{k=1}^{T} \gamma^{k-1} r_{t+k}$, where r_t is 1 if a new vent is found on step t and 0 otherwise, and the discount factor γ was increased to 0.99 because otherwise the results were skewed by trials where a vent happened to be found very early in the trial.

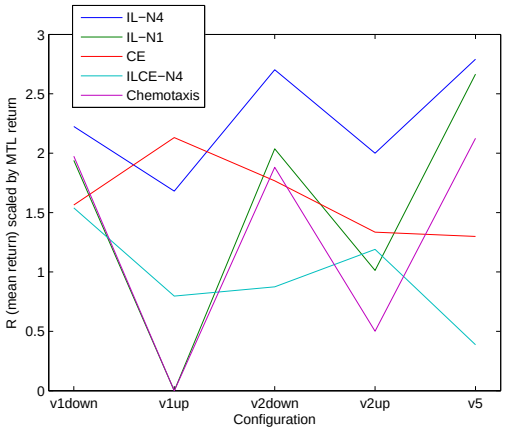
The results for μ_{vent} and R are shown in Figure 5 (a) and (b), where lines connect the data points for the same algorithm purely to aid visualization. IL-N4 is clearly the best performing algorithm, having the highest μ_{vent} values for all configurations, and the highest mean returns for four of the five configurations. IL-N1 and chemotaxis perform very similarly, which is interesting because despite being very different algorithms, they both rely on very local information for planning. IL-CE does unexpectedly badly in most cases; however, it is better than the two 'local' algorithms (chemotaxis and IL-N1) when the agent starts up-current of the vent(s), and information gain is important.

Figure 5 (c) and (d) show sample paths generated by the IL-N4 and chemotaxis algorithms. Given the available information, the IL-N4 agent seems explore the search area sensibly, and found at least four of the five vents in the 'v5' configuration on 86% of trials. Figure 5 (d) shows that the chemotaxis approach is also quite effective, as it explores a good proportion of the search area, albeit less efficiently than IL-N4.

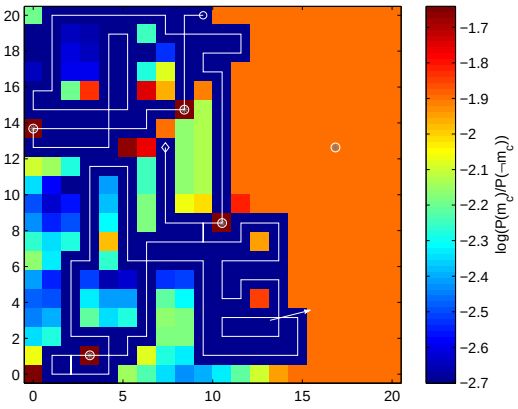
The execution times for one trial of 160 timesteps were as follows: IL-CE 4.0 hours, IL-N4 2.3 hours, IL-N1 15 seconds, CE 10 seconds, and chemotaxis and MTL both 4 seconds (on a 2.4Ghz Intel machine with 4 Gb RAM).



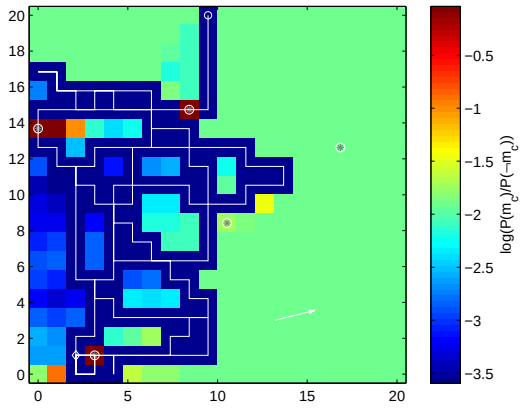
(a) μ_{vent}, the mean number of vents found. Note the lines connecting data points are purely to aid visualisation.



(b) R, mean of the total return from each trial. Values are scaled so the MTL result from each configuration is 1.0, and $\gamma = 0.99$ was used to calculate R.

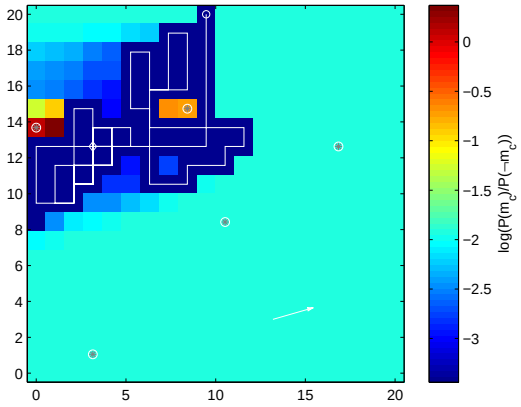


(c) Sample path taken by the IL-N4 algorithm, which was the most effective algorithm, after 160 time-steps. The path is shown in white against a background of the OG as estimated by the agent. The (unknown to the agent) locations of vents are plotted as filled circles, the start and end of the AUV's path are marked by a circle and a diamond respectively, and the white arrow at the bottom right is the average current.

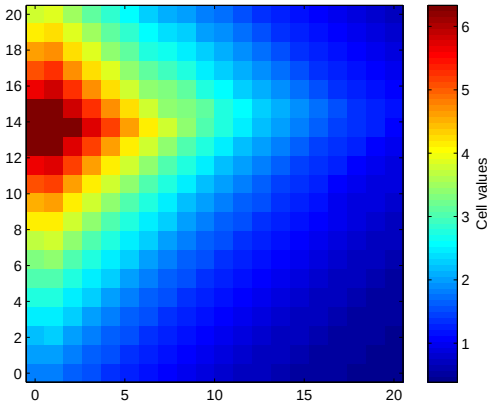


(d) Sample path taken by the chemotaxis algorithm, shown against the OG estimated by the agent, after 160 time-steps. Some of the zigzag lines are when the agent has been directed on a straight path to the centre of the grid, because it tried to leave the search area.

Figure 5: Results of simulated vent prospecting using 6 different algorithms and 5 different configurations of vents and starting position. All algorithms and configurations were evaluated over 50 trials, using a different random seed for each trial.



(a) Path produced by the IL-CE algorithm, against a background of the OG as estimated by the agent. The (unknown to the agent) locations of vents are plotted as filled circles, the start and end of the AUV's path are marked by a circle and a diamond respectively, and the white arrow at the bottom right is the average current.



(b) Certainty-equivalent cell values corresponding to the OG shown in subfigure (a). Note the maximum values are about 6, which is much more than the reward for finding a single vent, $R_{vent} = 1$

Figure 6: Snapshots of a particular trial of IL-CE algorithm with the 'v5' configuration with 5 vents, after 130 time-steps.

6 Discussion and Future Work

One of the challenges with this work is that we have no way to compute the optimal behaviour, even for relatively small occupancy grids. However, it is clear that information lookahead with a four step lookahead does best of all the algorithms we compared, but that even with a lookahead of one it still does surprisingly well, finding more vents than the other algorithms as the number of vents increases. An important feature of the IL algorithm is that although it is exponential in the number of steps of lookahead, it is independent of the size of the occupancy grid. The mapping algorithm is approximately linear in the number of cells, so there is no reason why this approach can't be scaled to much larger grids.

A more interesting result is that using the CE heuristic actually hurts information lookahead to such an extent that not only is IL-CE worse than IL with the simpler heuristic of the occupancy grid probability, but it is even worse than CE without any lookahead. This surprising result is due to the fact that when the CE value is computed, a reward is given when a state is revisited multiple times, so visiting high probability states looks more attractive during the CE phase than during IL because during IL the value of the state becomes zero once it has been visited. Hence, as Figure 6 shows, the IL-CE algorithm actually avoids high probability states, thinking it can get more value for visiting them later. To prevent revisiting states, we are investigating using a travelling salesman formulation to calculate the CE value rather than dynamic programming, but the computational cost is presently too high.

We are currently investigating other approaches to generating the heuristics. One issue is that on large grids four steps of lookahead may be insufficient to see useful areas to explore. A hierarchical approach that generates heuristics for finer-grained grids based on planning in more coarse-grained ones holds some promise here.

The current mapping algorithm assumes all grid cells are independent, whereas in reality, learning that there is a vent at some location should 'explain away' previous observations and therefore reduce the probability of nearby cells. Our planning algorithms should be agnostic about improving the mapping algorithm, but a better map may make a more significant performance improvement than better planning.

Finally, our current approach is entirely two-dimensional. Real AUV operations have to take the 3d

behaviour of the vent plume into account, and we plan in future to extend the approach to a more realistic 3d ocean model.

Acknowledgments

The authors would like to thank Michael Jakuba for generously providing us with code implementing his algorithms.

References

[Bresina *et al.*, 2002] J. Bresina, R. Dearden, N. Meuleau, S. Ramkrishnan, D. Smith, and R. Washington. Planning under continuous time and resource uncertainty: A challenge for AI. In *Proc. of UAI-02*, pages 77–84. Morgan Kaufmann, 2002.

[Darrell, 1997] T. Darrell. Reinforcement learning of active recognition behaviors. Technical Report 1997-045, Interval Research, 1997.

[Dearden *et al.*, 2007] R. Dearden, Z. Saigol, J. Wyatt, and B. Murton. Planning for AUVs: Dealing with a continuous partially-observable environment. In *Workshop on Planning and Plan Execution for Real-World Systems, 17th ICAPS*, 2007.

[Elfes, 1989] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[Farrell *et al.*, 2003] J. Farrell, S. Pang, W. Li, and R. Arrieta. Chemical plume tracing experimental results with a REMUS AUV. *Oceans Conference Record (IEEE)*, 2:962–968, 2003.

[Feng *et al.*, 2004] Z. Feng, R. Dearden, N. Meuleau, and R. Washington. Dynamic programming for structured continuous Markov decision problems. In *Proc. of UAI '04*, pages 154–161, 2004.

[Hauskrecht and Kveton, 2004] M. Hauskrecht and B. Kveton. Linear program approximations for factored continuous-state Markov decision processes. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in NIPS 16*. MIT Press, 2004.

[Jakuba and Yoerger, 2008] M. Jakuba and D. Yoerger. Autonomous search for hydrothermal vent fields with occupancy grid maps. In *Proc. of ACRA'08*, 2008.

[Jakuba, 2007] M. Jakuba. *Stochastic Mapping for Chemical Plume Source Localization with Application to Autonomous Hydrothermal Vent Discovery*. PhD thesis, MIT and WHOI Joint Program, February 2007.

[Kollar and Roy, 2008] T. Kollar and N. Roy. Efficient optimization of information-theoretic exploration in SLAM. In *AAAI '08*, pages 1369–1375, 2008.

[Littman *et al.*, 1995] M. Littman, A. Cassandra, and L.Kaelbling. Learning policies for partially observable environments: scaling up. In *Proc. 12th ICML*, pages 362 – 70, 1995.

[Martin and Moravec, 1996] M. Martin and H. Moravec. Robot evidence grids. Technical Report CMU-RI-TR-96-06, CMU Robotics Institute, 1996.

[Martinez-Cantin *et al.*, 2007] R. Martinez-Cantin, N. de Freitas, A. Doucet, and J.A. Castellanos. Active policy learning for robot planning and exploration under uncertainty. In *RSS*, 2007.

[Mausam *et al.*, 2005] Mausam, E. Benazera, R. Brafman, N. Meuleau, and E. Hansen. Planning with continuous resources in stochastic domains. *Proc. of the 19th IJCAI*, 19:1244, 2005.

[Paquet *et al.*, 2005] Sebastien Paquet, Ludovic Tobin, and Brahim Chaib-Draa. An online POMDP algorithm for complex multiagent environments. In *Proc. of AAMAS '05*, 2005.

[Poupart and Boutilier, 2004] P. Poupart and C. Boutilier. VDCBPI: An approximate scalable algorithm for large scale POMDPs. In *Advances in NIPS 17*, pages 1081–1088, 2004.

[Ross *et al.*, 2008] Stephane Ross, Joelle Pineau, Sebastien Paquet, and Brahim Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[Russell *et al.*, 2003] R.A. Russell, A. Bab-Hadiashar, R. Shepherd, and G. Wallace. A comparison of reactive robot chemotaxis algorithms. *Rob. Aut. Sys.*, 45(2):83–97, November 2003.

[Sridharan *et al.*, 2008] M. Sridharan, J. Wyatt, and R. Dearden. HiPPo: Hierarchical POMDPs for planning information processing and sensing actions on a robot. In *Proc. of ICAPS '08*, 2008.

[Thrun *et al.*, 2005] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.