Efficient optimization of information-theoretic exploration in SLAM

Thomas Kollar and **Nicholas Roy**
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
The Stata Center, 32 Vassar St.
Cambridge, MA 02139
{tkollar||nickroy}@mit.edu

Abstract

We present a novel method for information-theoretic exploration, leveraging recent work on mapping and localization. We describe exploration as the constrained optimization problem of computing a trajectory to minimize posterior map error, subject to the constraints of traveling through a set of sensing locations to ensure map coverage. This trajectory is found by reducing the map to a skeleton graph and searching for a minimum entropy tour through the graph. We describe how a specific factorization of the map covariance allows the reuse of EKF updates during the optimization, giving an efficient gradient ascent search for the maximum information gain tour through sensing locations, where each tour naturally incorporates revisiting well-known map regions. By generating incrementally larger tours as the exploration finds new regions of the environment, we demonstrate that our approach can perform autonomous exploration with improved accuracy.

Introduction

Computing the best trajectories for a mobile robot exploring an unknown environment typically requires satisfying the two competing objectives of completeness and accuracy. Firstly, the exploration trajectory must choose measurements that cover the environment, ensuring that the robot has collected measurements of the entirety of the environment in order to build a complete map. Secondly, the exploration trajectory must choose measurements that allow the SLAM algorithm to correctly infer relationships between different parts of the environment. Generating the most accurate map requires maximizing the mutual information between different environmental features and minimizing the loss of of information due to the robot motion.

Different pieces of this problem have been examined in previous work (Yamauchi 1997; Feder, Leonard, & Smith 1999; Makarenko *et al.* 2002; Stachniss & Burgard 2003; Sim & Roy 2005; Stachniss 2006), but a consistent optimization process for addressing these two problems remains an open problem. In particular, there has been little work on performing multistep destination selection, primarily due to the fact that computing even a standard minimum-distance trajectory is NP-hard. The problem is further complicated by the fact that the objective functions of information gain and expected error do not obey the triangle inequality, preventing the use of many common optimization techniques.

In this paper, we describe an algorithm for computing an exploration trajectory of an unknown environment by combining a global optimization of a tour of environment waypoints with a local controller that optimizes the trajectory through the selected waypoints. The primary contribution of our paper is a local-global optimization algorithm that uses hill climbing to optimize a coverage tour through a partially mapped environment and a learned controller that predicts the trajectory between waypoints that maximizes the information gain. Our second contribution is to show that the hill climbing can be performed efficiently using an alternate form of standard SLAM process-measurement updates.

SLAM with the EKF

It is beyond the scope of this paper to summarize the state of the art in robotic mapping; our overall approach to exploration is not specific to the map representation or inference algorithm, although we are able to achieve substantial gains in efficiency by assuming Gaussian map and robot posteriors. Without loss of generality, we will restrict our discussion to Extended Kalman filter SLAM.

Let us denote the robot position at time t as $\mathbf{x}_t = (x_t, y_t, \theta_t)$, and each map feature or "landmark" as $m_i = (x, y)$. The complete map of n landmarks is described as $\mathbf{m} = \{m_1, \ldots, m_n\}$. When the robot executes a control action $\mathbf{u}_t$, its next position is given by a process model $\mathbf{x}_{t+1} = g(\mathbf{x}_t, \mathbf{u}_t, w_t)$ where w_t is some unknown random noise that is added to the effect of the control. After each motion, the robot receives observations according to its current position and the landmarks. The observation of landmark i is given by a measurement model $z_i = h(\mathbf{x}, m_i, v_t)$ where v_t is some unknown random noise that is added to the received measurement. Typically a set of observations $\mathbf{z}_t$ is received at time t.

The goal of a SLAM algorithm is to infer the robot pose and the map based on knowledge of the control actions and observations. By filtering the actions and observations, a joint distribution over the robot pose and the map can be maintained, specifically the distribution $p(\mathbf{x}_t, \mathbf{m}|\mathbf{u}_{1:t}, \mathbf{z}_{1:t})$. If we assume this distribution is Gaussian, then the posterior is parametrized by the mean μ_t and covariance Σ_t statistics. If the process and measurement models are linear functions and the unobservable noise terms are Gaussian, then the mean of the posterior distribution computed

Algorithm 1 The Extended Kalman Filter

```
Require: Prior mean and covariance $(\mu_{t-1}, \Sigma_{t-1})$, control
    $\mathbf{u}_t$ and observation $\mathbf{z}_t$.
1: Compute process mean,
       $\overline{\mu}_t = g(\mu_{t-1}, \mathbf{u}_t, 0)$
2: Compute process Jacobian $G_t|_{\mu_{t-1}}$.
3: Compute process covariance,
       $\overline{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$.
4: Compute measurement Jacobian $H_t|_{\overline{\mu}_{t-1}}$.
5: Compute Kalman gain,
       $K_t = \overline{\Sigma}_t H_t^T \left( H_t \overline{\Sigma}_t H_t^T + Q_t \right)^{-1}$.
6: Compute new mean,
       $\mu_t = \overline{\mu_t} + K_t(h(\overline{\mu}_t, 0) - z_t)$.
7: Compute new covariance,
       $\Sigma_T = (I - K_T H_T)\overline{\Sigma}_T$.
```

by the Kalman filter can be shown to minimize the least-squares error. When the models are non-linear, the extended Kalman filter can be used to repeatedly linearize the process and measurement models. The Jacobians of the process and measurement models g and h used to linearize these models are G and H respectively, and R and Q are the covariances of the Gaussian noise distributions for the process and measurement updates. The complete Extended Kalman Filter algorithm is shown in figure 1.

Exploration

Given sufficient data, the mean estimate of the map provided by the EKF will usually converge to the true map[1]. Unfortunately, few if any guarantees are given for the rate at which the map converges; the quality of the map is highly dependent on the robot trajectory as well as a variety of other features such as sensor model, feature distinctiveness for data association, etc..

The goal in this paper is to choose trajectories that lead to sensor data that results in the best map, maximizing both the map coverage and the map accuracy. There are different ways to reconcile these objectives, such as optimizing the robot trajectory with respect to an overall objective that is a weighted sum of individual terms (Makarenko *et al.* 2002), but such approaches raise additional questions such as how to choose the individual weights. Our approach is to define the coverage problem as choosing the best sensing locations that maximize the likelihood the complete environment will be observed. If these sensing locations serve as constraints on a trajectory planner, and we compute a trajectory that maximizes the map accuracy subject to these constraints, then we have a principled way to optimize both coverage and accuracy. The trajectory constraints based on coverage can be calculated first, and then the constrained trajectory can be calculated to maximize map accuracy.

[1]Although, convergence guarantees only hold for the standard Kalman filter, not the extended form.

Map Coverage

Let us first consider the problem of coverage. We define a "visibility" function $\delta(m^i, \mathbf{z}_{0:T})$,

$$\delta(m^i, \mathbf{z}_{0:T}) = \begin{cases} 1 & \text{if map feature } m^i \text{ appears at least once in the observations } \mathbf{z}_{0:T}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The goal of a coverage planner is to search for the trajectory that maximizes the number of sensed features,

$$\operatorname*{argmax}_{\mathbf{z}_{0:T}} \sum_{i=1}^{|\mathbf{m}|} \delta(m^i, \mathbf{z}_{0:T}). \quad (2)$$

We cannot control or predict *a priori* what observations $\mathbf{z}_t$ will be received. We can control the robot poses $\mathbf{x}_{0:T}$, but the observations are generated stochastically according to the map and the robot pose, $p(\mathbf{z}_{0:T}|\mathbf{x}_{0:T}, \mathbf{m})$. We do not know the map $\mathbf{m}$, so equation (2) must be computed in expectation over the map prior distribution $p(\mathbf{m})$ and observation likelihoods, such that

$$\operatorname*{argmax}_{\mathbf{x}_{0:T}} E_{p(\mathbf{m})} E_{p(\mathbf{z}_{0:T}|\mathbf{x}_{0:T},\mathbf{m})} \left[\sum_{i=1}^{|\mathbf{m}|} \delta(m^i, \mathbf{z}_{0:T}) \right]. \quad (3)$$

The distribution $p(\mathbf{m})$ is actually conditioned on the measurements up to the current time, which have been left out for brevity.

Computing the double expectation exactly over both map prior and observations is computationally infeasible, but the distribution over maps has only limited impact on the optimization. At the beginning of the mapping process, the map prior will be very uncertain, which will cause all trajectories to have approximately the same expected performance, as all maps are equally likely. As more observations are made, the map distribution will begin to converge to the true map, and the mean map estimate $\hat{\mathbf{m}}$ will become an increasingly good approximation to the complete distribution. When the map prior is informative, Monte Carlo methods may be used to estimate the outer expectation directly. Thus, we can approximate equation (3) as:

$$\operatorname*{argmax}_{\mathbf{x}_{0:T}} E_{p(\mathbf{z}_{0:T}|\mathbf{x}_{0:T},\hat{\mathbf{m}})} \left[\sum_{i=1}^{|\mathbf{m}|} \delta(m^i, \mathbf{z}_{0:T}) \right]. \quad (4)$$

While equation (4) is an approximation to our true coverage objective, in practice a robot coverage algorithm that chooses a set of destination points to maximize the likelihood of completing the map based on the current estimate, and then follows a tour through these points will have performance comparable to any other sequential decision making algorithm.

We can gain additional computational tractability in solving equation (4) by considering only a subset of possible locations of the robot, that is, a retraction of the map free space. Map features are constrained (by definition) to lie on the boundary of free space, so our retraction must be a subset of map locations that guarantee that every boundary point can be observed from some point in this subset.

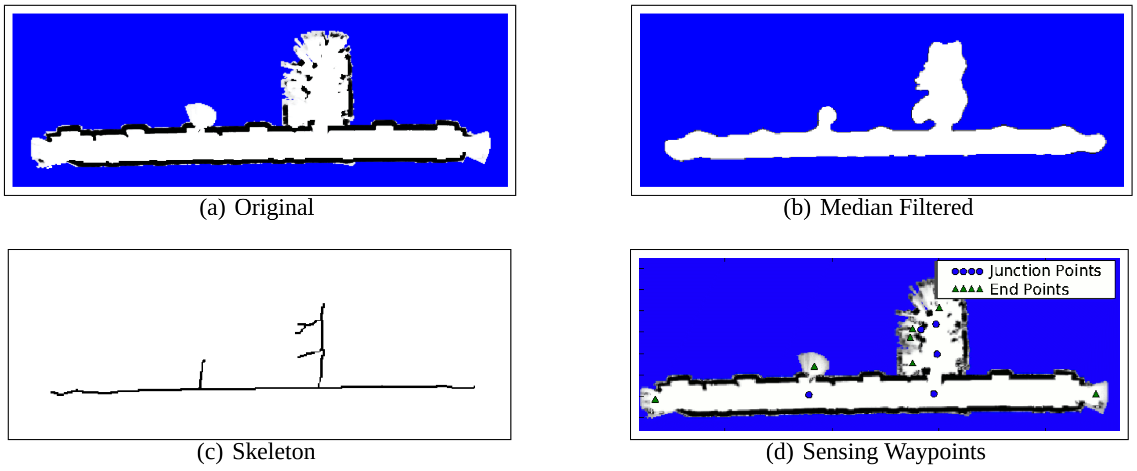


Figure 1: The map skeletonization process. (a) The binarized map. (b) The smoothed map using median filtering to eliminate high-frequency noise. (c) The piece-wise linear graph of the medial-axis skeleton. (d) The graph nodes that constitute sensing location constraints for the trajectory optimization.

This property of boundary visibility is known as “reliability” (Cornea, Silver, & Min 2005); using the medial-axis transform (Blum 1967) to reduce the free-space to a minimal “skeleton” allows us to approximate the map with a minimal set of points while guaranteeing reliability necessary for equation (4). Efficient algorithms for computing a medial axis representation that guarantee reliability generally lead to near-optimal reductions in terms of minimizing the resulting skeleton; computing the true minimum medial axis representation is NP-hard (Coeurjolly, Hulin, & Sivignon 2007).

In order to select a set of sensing locations that provide a guarantee of coverage, we retract the map to its medial axis. We then further reduce the skeleton to a graph by converting the resulting map skeleton to a piece-wise linear representation, retaining only the end-points of the resulting lines. We can then use graph search algorithms to identify a tour of the skeleton that maximizes the predicted map accuracy. By generating a graph with guaranteed boundary visibility and constraining our exploration trajectory to traverse the graph nodes, the exploration trajectory is constrained to have full map coverage.

The complete skeletonization algorithm is given by Algorithm 2. Although the map free space is given by an occupancy grid, this grid is binarized before skeletonization. Note also that high frequency objects in the map such as discretization artifacts or small mapping errors can create an overabundance of end-points and junction points. To reduce the effect of high-frequency areas to have less influence on the end-points and junction points, we first median filter the image, as shown in figure 1(b).

Algorithm 2 Map Skeletonization

```
Require: Map $m$.
1: Median filter the binarized map.
2: Skeletonize the map using the medial-axis transform.
3: Reduce the skeleton to a piece-wise linear graph.
4: Return the end-points of lines (junctions and graph
   leaves).
```

Map Accuracy

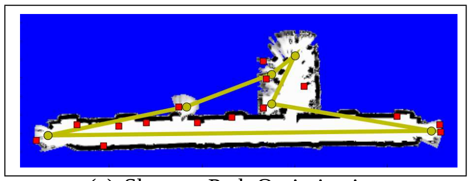
Given a graph through the environment that maximizes the expected coverage of map features, we wish to compute a tour through these points, that is, an ordering on the graph points, to maximize the resulting map accuracy. We cannot measure the map accuracy without ground truth, but the covariance of the posterior distribution provides an estimate of the expected squared error. Minimizing the determinant of the covariance of the posterior Gaussian is equivalent to minimizing the differential entropy described by Darbellay & Vajda (2000) as

$$H(p(x)) \quad = \quad \frac{1}{2}\ln[(2\pi e)^n|\Sigma|\,]$$

As a result, computing a tour that minimizes the posterior map entropy is equivalent to an information-theoretic traveling salesman problem where the objective function is information maximization rather than distance minimization. Unfortunately, as with most traveling salesmen problems, this optimization is again NP-hard (Garey & Johnson 1979). Furthermore, the marginal gains for computing the globally optimal trajectory are likely to be low. On the other hand, local optimization will allow the planner to take advantage of some information gathering actions such as revisiting known-locations that a greedy strategy will not always use.

As a result, we use local optimization similar to the gradient-ascent heuristics used for solving shortest-path TSP problems, such as the edge-swapping heuristic. A basic tour is generated from a simple greedy nearest-neighbor heuristic. At each iteration two edges are swapped and the information gain is calculated. If the posterior map entropy is improved, then this edge move is retained. Edge-swaps are continued until no more edges are available to reduce the tour cost, and the best (lowest predicted posterior entropy) tour is used.

Note that for the conventional shortest-path TSP, this approach is referred to as the 2-opt heuristic and is known to be better than a $O(log(n))$ approximation to the true distance-


(a) Shortest Path Optimization

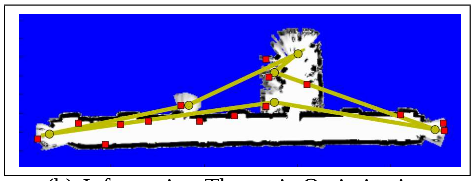
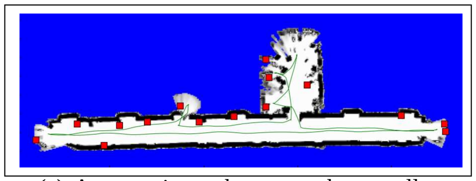

(b) Information-Theoretic Optimization

Figure 2: Optimized tours. (a) For the graph in figure 1, the ordering of points that minimizes the total path length. $(H(p(\mathbf{m})) = -6.93)$ ((b) For the same graph, the ordering of points found by minimizing the posterior map entropy. $(H(p(\mathbf{m})) = -7.23)$. (Note that the red lines only show connectivity for clarity of presentation. The path lengths are calculated through free space, but the paths tend to overlap and obscure the resulting connectivity.)


(a) A tour using a shortest-path controller

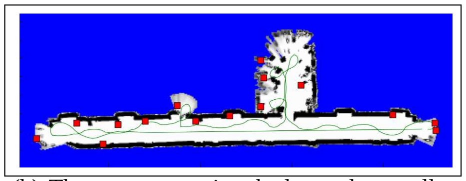

(b) The same tour using the learned controller

Figure 3: Comparison of a path executed by a conventional shortest-path controller that uses point turns (a) with a path executed by the learned controller. The tour of the graph in both cases was fixed.

optimal solution. This performance guarantee does not hold for the exploration problem because information gain violates the triangle inequality property (Rosenkrantz, Stearns, & Lewis 1977), but performs well in practice.

Figure 2 shows the results of using the 2-opt heuristic with the shortest path objective (figure 2a) and the information-theoretic objective (figure 2b). The principal difference between the planned tours is the order of points at the top of the free space; by first taking a straight trajectory into the room, the information-theoretic optimization improves the map of the room and then revisits the remaining locations, using the improved map to stay well-localized. Additionally, the information-theoretic optimization uses waypoints in previously-visited locations to relocalize. In contrast, the shortest-path trajectory visits the early waypoints first and cannot deliberately revisit a location to re-localize, resulting in a lower-quality map. The information-theoretic optimization significantly reduces the map entropy, and as expected also reduces the squared error of the resulting map (table 1).

One-step update

In order to evaluate the posterior map entropy of a particular exploration trajectory, we can use the EKF algorithm to predict the posterior map estimate after the trajectory. This prediction requires computing the expected motion updates, simulating measurements from the current map estimate and updating the EKF at each time step. As a result, the prediction process can be computationally demanding and the cost is intensified by the fact that every modification of the exploration trajectory requires the EKF to recalculate the map for the entire trajectory following the modification. Even if the trajectory modification is relatively minor, the non-linearity of the covariance update in Algorithm 1 means that the EKF updates from one trajectory cannot easily be re-used in the overlapping section of another trajectory. Note that the simulated motions and measurements can be re-used if the same graph edge appears in two potential trajectories; it is only the EKF updates along that must be recalculated.

In previous work (Prentice & Roy 2007) we have shown that multiple updates to a Gaussian filter can be combined into a single transfer function by re-factoring the covariance, allowing multiple process and measurement updates to occur with a constant number of operations. If the covariance is factored as $\Sigma_{t-1} = B_{t-1}C_t^{-1}$, then the individual factors can be updated as

$$\begin{bmatrix} B \\ C \end{bmatrix}_t = \begin{bmatrix} W & X \\ Y & Z \end{bmatrix}_t \begin{bmatrix} B \\ C \end{bmatrix}_{t-1} \tag{5}$$

$$= \begin{bmatrix} 0 & I \\ I & M \end{bmatrix}_t \begin{bmatrix} 0 & G^{-T} \\ G & RG^{-T} \end{bmatrix}_t \begin{bmatrix} B \\ C \end{bmatrix}_{t-1}, \tag{6}$$

and Σ_t recovered as $\Sigma_t = B_t C_t^{-1}$. (Note that $M = H_t^T Q^{-1} H_t$, the measurement information.) Assuming we have known measurements and motions, we recursively apply the measurement Jacobian H, the measurement noise Q, the motion Jacobian G, and the motion noise R in the form of the transfer function $\zeta_t = \left[\begin{smallmatrix} W & X \\ Y & Z \end{smallmatrix}\right]_t$ to the original $\left[\begin{smallmatrix} B \\ C \end{smallmatrix}\right]_0$. We can initialize the process as $B_0 = \Sigma_0$ and $C_0 = I$, where Σ_0 is the initial covariance matrix and I, the identity matrix. Note that each of these terms can be seen in the original formulation of the EKF (Algorithm 1).

Using this one-step EKF update, we can collect simulated motions and measurements along each edge in the graph, compute the relevant Jacobians for each time step and collect terms into a single transfer function for that edge that can be reused for the map posterior prediction of future exploration trajectories. In particular, we compute a tour by creating one-step transfer functions between the interest points. If we have already created a transfer function to within some threshold ϵ of the original orientation, we will use the pre-generated transfer function to obtain the one-step update. Thus, we will cache the transfer functions for any already-visited edges and use this to drastically speed up the search process.

Motion Control Optimization

The coverage exploration optimization assumes a specific model of robot motion along each edge in the graph but does not have direct control of the robot motion between sensing locations. However, the trajectory between sensor measurements can have a significant effect on the accuracy of the map. When the robot motion is noisy, substantial uncertainty is introduced into the robot pose; when few or no observations are expected (e.g., because of sensor range limits or environmental sparsity), this uncertainty is propagated throughout the SLAM filter into future measurements, increasing the expected error of the posterior.

To learn good control policies, we use Policy Search Dynamic Programming (Bagnell *et al.* 2003; Kollar & Roy 2008), a form of reinforcement learning that decomposes into a series of one-step learning algorithms, essentially turning the learning of sequential decision making into dynamic programming. The algorithm operates by first learning a one-step policy at time T (e.g., the end time). Training data is generated by sampling a robot pose and map distribution $p(\mathbf{x}, \mathbf{m})$, and using all possible control trajectories a to propagate each sampled prior distribution forward to a posterior distribution according to the EKF using a physically realistic vehicle and sensor simulator. Each distribution-action pair is labeled with the resulting information gain, and the distribution-action pair that minimizes the map entropy is kept as a training instance for a supervised learning problem at time t. The result is a classifier that provides an action a for any distribution $p(s)$ that minimizes the map entropy. A one-step policy for the previous time step $T-1$ is then learned by sampling distributions and actions as before, but propagating each distribution according to each action a followed by the time T one-step policy previously obtained. A two-step entropy is obtained for each distribution and action at $T-1$ from the resulting distribution-action pair: the entropy associated with each distribution and action is accumulated from the motion and observations associated with the control at time $T-1$ *and* the motion and observations received by running the learned policy for time T. A new policy is learned for time $T-1$, and the learner then iterates at each time t through to time T using the policies π_i for $i \in \{t \ldots T\}$, resulting in a motion controller that minimizes the entropy over the length of the trajectory $t = 0 \ldots T$.

Figure 3 compares a standard shortest-path motion controller with the learned controller for the same tour of sensing locations. As expected, the map entropy is significantly reduced for the learned controller ($H(m) = -7.23$) compared to the shortest-path controller ($H(m) = -6.93$).

Incremental Exploration

To summarize, our goal has been to optimally explore unknown environments by solving a local-global optimization problem, maximizing both map coverage and map accuracy. The coverage requirement has led us to take the partial map, skeletonize it, and extract sensing locations from it, as seen in Algorithm 2. The map accuracy requirement has led us to a hill-climbing algorithm that optimizes the visitation of sensing locations while at the same time minimizing the resulting entropy of the map. The initial tour is optimized with respect to distance, the 2-opt heuristic modifies the tour and the best tours are kept, as seen in Algorithm 3. Computing the EKF updates for previously visited tour edges is inefficient and can be optimized by storing a transfer function for each edge. Additional improvement can be achieved by using a local learned controller to plan local trajectories between sensing locations. Once the optimal tour is computed and then executed by the robot, the partial map of the environment will grow. When a complete tour is accomplished, a new tour will be computed for the new partial map and the next round of exploration will begin. In this section, we have presented a complete algorithm for information-theoretic exploration and have elaborated a number of ways that it can be made more efficient.

Algorithm 3 Information-theoretic tour optimization

```
Require: Approximate shortest-path TSP tour $t_{\text{best}}$ through
    sensing locations derived from skeletonized map
 1: while not converged do
 2:     Use the 2-opt heuristic to create a new tour $t_{\text{curr}}$, initialize $\Sigma_0$.
 3:     for each edge $e_{ij}$ in $t_{\text{curr}}$ do
 4:         if Edge $e_{ij}$ was previously executed starting from orientation $\theta_k$ then
 5:             Use the cached transfer function $\zeta_{ijk}$ to update $\Sigma_t$ to $\Sigma_{t+n}$.
 6:         else if $e_{ij}$ has not been executed before then
 7:             Execute it using the learned controller and store $\zeta_{ijk}$ for that edge and initial orientation.
 8:         end if
 9:     end for
10:     Compute the entropy $H_{\text{curr}} = \text{entropy}(\Sigma_T)$ after executing tour $t_{\text{curr}}$.
11:     if $H_{\text{curr}} < H_{\text{best}}$ then
12:         $t_{\text{best}} = t_{\text{curr}}$.
13:     end if
14: end while
15: return $t_{\text{best}}$
```

Results

To demonstrate the effectiveness of the approach, we tested the exploration algorithm on scenarios from the Radish dataset using a simulation with uncertain robot motion and uncertain landmark measurements. Treating both the WEAN and LONGWOOD maps as partial maps, we ran the optimization with the shortest path controller and with the learned controller. Overall, it is clear that the map entropy after exploration has been reduced, and in all but the one instance (the WEAN map with the information-theoretic objective and the learned controller), the squared error is also reduced for the optimized trajectory. In the larger LONGWOOD example (Figure 4), it appears clear that the tour stays in sight of as many map features (red squares) as possible.

Having shown that we can successfully reduce the map entropy and improve map accuracy, we next demonstrate an incremental exploration of the environment in Figure 5. Here the robot has a partial map of the environment, skeletonizes this map to find sensing locations, computes an information-theoretic tour, and then executes that tour. After the execution of the tour in (d), the robot then computes a new tour, which it then executes in (e), exploring this partial map as well until the entire environment has been observed (f). Table 2 gives a comparison of the entropy and map accuracies for the maps in Figure 5 corresponding to the distance optimal and information-theoretic exploration algorithms.

Noting that each pair of maps (a-d, b-e, c-f) may be di-

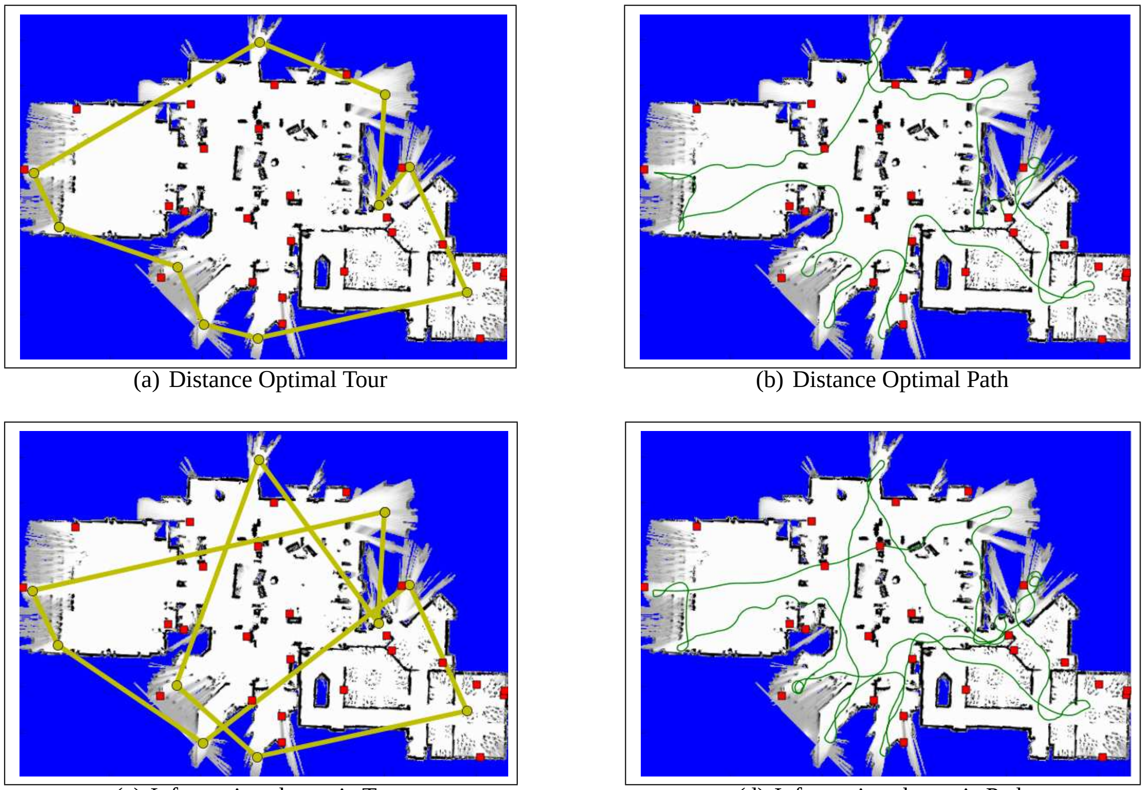

(a) Distance Optimal Tour (b) Distance Optimal Path (c) Information-theoretic Tour (d) Information-theoretic Path

Figure 4: Tours for the Longwood map. (a) The shortest path ordering of the skeleton map graph. (b) The resulting path using the shortest-path controller. (c) The minimum-entropy path ordering of the skeleton map graph. (d) The resulting path using the learned controller.

MAP	OPT. TYPE	CONT.	ENT.	SE	DIST.
W	Path Lgth.	Dist. Opt	-6.93	0.041	107.8m
W	Info	Dist. Opt	-7.23	0.033	112.5m
W	Info	Learned	-7.9	0.13	125.3m
L	Path Lgth.	Learned	-5.33	2.01	238.3m
L	Info	Learned	-6.99	0.45	302.7m

Table 1: Algorithm Performance for a Single Tour. MAP is the map that the experiment was performed in, W is the WEAN map and L is the LONGWOOD map. OPT. TYPE corresponds to graph tour optimization, either "shortest path" or "information-theoretic optimization". CONT. corresponds to the motion controller, either "shortest path" or "learned". ENT. and SE correspond to the metrics of map entropy and map squared error compared to the known ground truth respectively. DIST. corresponds to the distance the robot traveled over the entire tour.

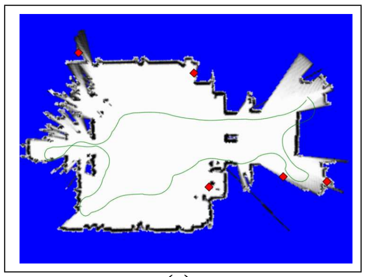
	OPT.	CONT.	ENT.	SE	DIST
(a)	Dist	Learned	-1.82	0.043	70.7m
(b)	Dist	Learned	-3.33	2.16	95.5m
(c)	Dist	Learned	-4.08	0.358	139.0m
(d)	Info	Learned	-2.54	0.040	77.9m
(e)	Info	Learned	-2.95	0.03	136.2m
(f)	Info	Learned	-5.09	0.028	177.7m

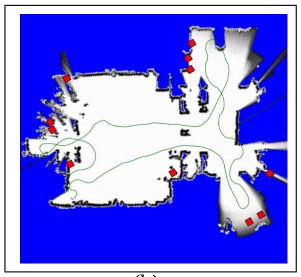
Table 2: Algorithm for a sequence of tours during online exploration. OPT. corresponds to graph tour optimization, either "shortest path" or "information-theoretic optimization". The controller CONT. in every case was the learned controller . ENT. and SE correspond to the metrics of map entropy and map squared error respectively. DIST. corresponds to the distance the robot traveled over the entire tour. The information-theoretic tour optimization results in both a reduced map entropy and reduced map error compared to the known ground truth.

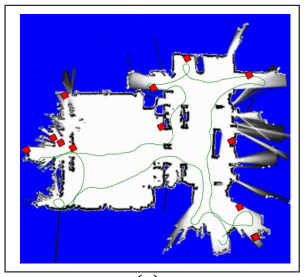
rectly compared, we found that the squared error and the entropy were both significantly reduced by the optimized policy. Further, by using one-step transfer functions, we were able to reduce the total number of edge prediction steps by 50%, for a similar time savings of 50%. We expect this savings will grow for larger problem sizes.

Conclusion

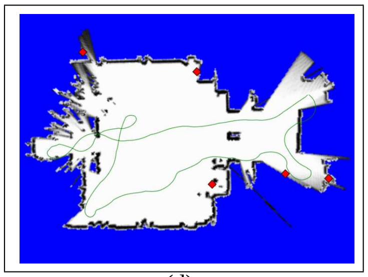
We have presented a novel method that frames exploration as a constrained optimization problem. The algorithm reduces the current environmental map to a graph of the map skeleton, placing sensing constraints at the graph nodes which naturally include boundaries and frontiers. Information-theoretic tours are then generated through the sensing constraints using a gradient-ascent with a heuristic; we can perform the optimization efficiently by using one-step transfer functions on the covariance to re-use previously computed EKF updates. To our knowledge this is one of few information-theoretic approaches to exploration and the only one to consider multistep *tours* of an environment.

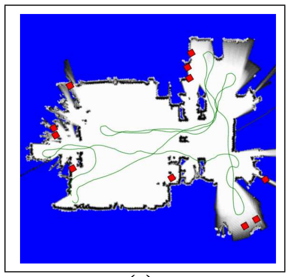

(a)

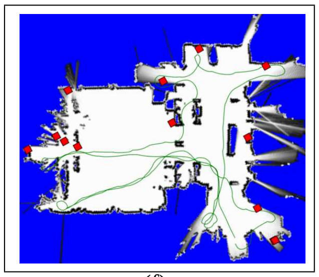

(b)


(c)

(a-c) Map exploration using shortest path optimization of exploration trajectory


(d)


(e)


(f)

(d-f) Map exploration using information-theoretic optimization of exploration trajectory

Figure 5: Example exploration trajectories, comparing a shortest-path coverage trajectory with an information-theoretic coverage trajectory. In panels (a) and (d), the algorithm has only a limited view of the environment, and plans a tour through the visible free space. In doing so, the map is expanded (b and e), and the robot can plan a second tour through the larger space. The process iterates again through a yet larger map in (c and f).

Finally, a learner is employed to optimize the local trajectory of the robot between sensing constraints, to obtain global information-theoretic exploration that can quickly find information-theoretic trajectories through an environment, resulting in a high quality map.

References

Bagnell, J.; Kakade, S.; Ng, A.; and Schneider, J. 2003. Policy search by dynamic programming. In *Neural Information Processing Systems*, volume 16. MIT Press.

Blum, H. 1967. *A Transformation for Extraction New Descriptors of Shape, Models for the Perception of Speech and Visual Form*. Cambridge: MIT Press.

Coeurjolly, D.; Hulin, J.; and Sivignon, I. 2007. Finding a Minimum Medial Axis of a Discrete Shape is NP-hard. Technical Report RR-LIRIS-2007-026, LIRIS UMR 5205 CNRS/INSA de Lyon/Universit Claude Bernard Lyon 1/Universit Lumire Lyon 2/Ecole Centrale de Lyon.

Cornea, N. D.; Silver, D.; and Min, P. 2005. Curve-skeleton applications. In *Proceedings IEEE Visualization*, 95–102.

Darbellay, G. A., and Vajda, I. 2000. Entropy expressions for multivariate continuous distributions. *IEEE Transactions on Information Theory* 46(2):709–712.

Feder, H. J. S.; Leonard, J. J.; and Smith, C. M. 1999. Adaptive mobile robot navigation and mapping. *International Journal of Robotics Research, Special Issue on Field and Service Robotics* 18(7):650–668.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. Series of Books in the Mathematical Sciences. W. H. Freeman.

Kollar, T., and Roy, N. 2008. Trajectory optimization using reinforcement learning for map exploration. *The International Journal of Robotics Research* 27(2):175–196.

Makarenko, A.; Williams, S.; Bourgault, F.; and Durrant-Whyte, H. 2002. An experiment in integrated exploration. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

Prentice, S., and Roy, N. 2007. The belief roadmap: Efficient planning in linear pomdps by factoring the covariance. In *Proceedings of the 13th International Symposium of Robotics Research (ISRR)*.

Rosenkrantz, D.; Stearns, R.; and Lewis, P. 1977. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing* 6:563.

Sim, R., and Roy, N. 2005. Global a-optimal robot exploration in slam. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Stachniss, C., and Burgard, W. 2003. Exploring unknown environments with mobile robots using coverage maps. In *Proc. of the International Conference on Artificial Intelligence (IJCAI)*.

Stachniss, C. 2006. *Exploration and Mapping with Mobile Robots*. Ph.D. Dissertation, University of Freiburg, Department of Computer Science.

Yamauchi, B. 1997. A frontier-based approach for autonomous exploration. *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation* 146–151.