2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Path and Trajectory Diversity: Theory and Algorithms

Michael S. Branicky* Ross A. Knepper† James J. Kuffner†

*EECS Department
Case Western Reserve University
10900 Euclid Avenue, Glennan 517B
Cleveland, OH, 44106, USA
mb@case.edu

†The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA, 15213, USA
{rak,kuffner}@ri.cmu.edu

***Abstract*— We present heuristic algorithms for pruning large sets of candidate paths or trajectories down to smaller subsets that maintain desirable characteristics in terms of overall reachability and path length. Consider the example of a set of candidate paths in an environment that is the result of a forward search tree built over a set of actions or behaviors. The tree is precomputed and stored in memory to be used online to compute collision-free paths from the root of the tree to a particular goal node. In general, such a set of paths may be quite large, growing exponentially in the depth of the search tree. In practice, however, many of these paths may be close together and could be pruned without a loss to the overall problem of path-finding. The best such pruning for a given resulting tree size is the one that maximizes *path diversity*, which is quantified as the probability of the survival of paths, averaged over all possible obstacle environments. We formalize this notion and provide formulas for computing it exactly. We also present experimental results for two approximate algorithms for path set reduction that are efficient and yield desirable properties in terms of overall path diversity. The exact formulas and approximate algorithms generalize to the computation and maximization of *spatio-temporal diversity* for trajectories.**

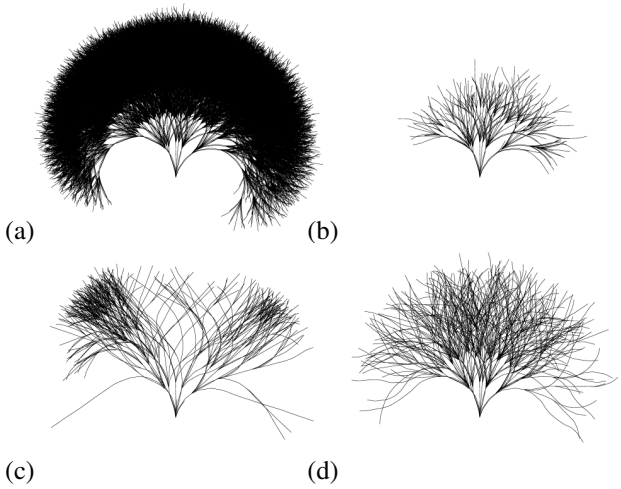

Fig. 1. Path sets: (a) The full 34,295-path data set; (b) The 1% subset generated by the Inner-Product algorithm; (c) The 1% subset generated by the Inclusion-Exclusion algorithm; (d) The 1% subset generated by random sampling from the full path set.

I. INTRODUCTION

Mobile robot applications most often impose time and resource constraints on their motion planning software. Real robotic systems replan at a fast rate in order to incorporate a constant stream of new perception data. Due to time constraints, most planners operate by considering a relatively small set of possible actions and commanding the best choice before the next execution deadline arrives. Examples of such systems include ground vehicles (driving on the highway at high speed or navigating through cluttered, off-road terrain), unmanned aerial vehicles including rotary and fixed-wing aircraft. Failure to generate a plan quickly and with sufficient lookahead risks damage to the vehicle. This concern is particularly acute in the case of fixed-wing aircraft, which do not possess the capability to stop in place if the planner misses its deadline.

One popular strategy for improving online planning time is to precompute a set of candidate actions a priori (e.g. [1], [2], [3], [4], [5]). The advantage of this is that no online computation time is wasted in considering paths that the robot cannot execute. The drawback is that a path set generated offline may not incorporate any knowledge of particular obstacle configurations.

Our work discusses the problem of evaluating a path or set of paths in the context of unknown obstacle location and density in the world. Two theoretical algorithms for computing an exact probability of collision are derived. Two approximate algorithms (Inner-Product and Inclusion-Exclusion) are then presented, which provide a tractable solution to the problem of selecting a diverse set of paths. These algorithms cull a large path set down to a smaller, more manageable size, while attempting to maximize the robustness of the path set

978-1-4244-1647-9/08/$25.00 ©2008 IEEE.

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.

(versus other path sets of the same size) with respect to unknown obstacle configurations; see Fig. 1. The choice of path set size for a particular application may depend on constraints such as planner requirements, available memory, and CPU performance. The reduced path sets are analyzed with respect to their survivability in the presence of obstacles.

A. Background

Building and storing discretized representations of state reachability and configuration space (C-space) volumes has been a major research thrust in robot motion planning. Popular sampling-based planning algorithms such as the probabilistic roadmap (PRM) [6] and Rapidly-exploring Random Trees (RRTs) [7] and their variants build graphs and trees of connected configurations for path planning in high-dimensional search spaces with obstacles. Some of the key research issues include developing good heuristics for placing samples, connecting pairs of samples, enforcing constraints, and exploring tradeoffs between what must be computed at runtime and what can potentially be precomputed (e.g. [8], [9]).

Leven and Hutchinson [2] precomputed a roadmap of samples in the configuration space to be stored in memory and used for efficient planning online. Although they did not consider the optimal placement of roadmap nodes, the network of paths they used and how well the roadmap covers the reachable configuration space is directly related to our problem of path diversity. In particular, our definition of path diversity has some of the same motivations as their notions of "ρ-robustness" and "edge robustness." Burns and Brock [10] explore techniques aimed at optimally placing samples for path planning using a machine learning framework. In this case, the focus was on quickly discovering narrow passages in the configuration space for a given problem during runtime. Ramamoorthy et al. [11] explore "low-discrepancy curves" that provide efficient, uniform coverage of configuration spaces (up to a dispersion error).

The idea of trading off memory for runtime performance in the context of path planning has become popular across a number of different fields. In the controls literature, storing sets of *trim trajectories* for helicopter dynamic models and the notion of composing them to form obstacle-avoiding paths has been investigated by Frazzoli [1]. Trim trajectories are essentially predefined maneuvers that are invariant to some state variables (such as the global translation and rotation). These invariant maneuvers are useful because they allow a high level planner to efficiently reason about local state reachability. For methods and a discussion of discretization and reachability issues in the context of continuous control systems, see Bicchi et al. [12]. In the computer animation literature, Go et al.[3] precompute a dense table of state-dependent trajectories that are affine invariant in order to avoid having to forward integrate the dynamics of simulated spacecraft during runtime. Lau and Kuffner [4] use precomputed trees of captured human motion data to efficiently plan animations for multiple characters interactively. In field robotics, the idea of maintaining multiple path alternatives for emergency stops or avoidance maneuvers has a fairly long history. Sets of steering paths were used for the robot that won the 2005 DARPA Grand Challenge for autonomous driving [5]. NASA's Mars rovers also utilize sets of steering curves for deciding local maneuvers [13].

Our work is most closely related to the work of Green and Kelly [14], which formulates the problem in terms of sampling in the continuous space of paths. Candidate paths are selected that are most likely to produce a solution given a probabilistic representation of all possible environments. Computing the optimal mutual separation between paths is shown to be related to the maximum *k*-facility dispersion problem which is known to be NP-hard.

II. PATH DIVERSITY

Given a set of paths, we say that a particular subset of paths (of a fixed size) maximizes *path diversity* if that subset is the most robust to the maintenance of feasible paths regardless of the placement of obstacles. Here, we measure such robustness by the probability of the maintenance of paths, averaged over all possible obstacle environments.[1] Therefore, in the case of pruning down a set of paths to a subset of a certain size, we wish to pick the subset that minimizes the probability of the failure (due to blocked paths). Below, we formalize this notion and provide formulas for computing it.

Let there be a finite set of paths $[\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, each of which lives in a space that has been partitioned into a set of cells $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$.

In this way, each path may be discretely represented by the sequence of cells through which it traverses.[2]

Given this setup, we pose the following two problems:

Problem 1 (Path Diversity Measure): Compute the probability (uniformly over all possible environments) that *at least one* of the paths in $\mathcal{P}$ is not blocked, denoted $P_{pnb}(\mathcal{P})$ [read this as "path-not-blocked"].

[1] In this paper, we consider all possible environments to be equally likely. However, the ideas generalize to arbitrary distributions.

[2] Actually for path diversity, it is only the *set* of cells traversed, and not their sequence that matters. The notion directly generalizes to trajectories by considering the sequence.

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.

Problem 2 (Maximal Path Diversity Pruning): Given $k < n$, find the subset $\mathcal{P}' \subset \mathcal{P}$ of size k such that $P_{pnb}(\mathcal{P}')$ is maximized.

Below, we make use of the following notation. $|A|$ denotes the cardinality of set A. Thus, $|\mathcal{P}| = n$ and $|\mathcal{C}| = m$. 2^A denotes the power set of set A. Note that $|2^{\mathcal{C}}| = 2^{|\mathcal{C}|} = 2^m$. We will also use the predicate notation of Knuth: $[\psi]$ equals 1 if the predicate ψ is true; 0, if it is false. We will use $P_{cnb}(A)$ [read this as "cells-not-blocked"] as a shorthand for $P_{pnb}(\{A\})$. In particular, $P_{cnb}(p_i)$ is shorthand for $P_{pnb}(\{p_i\})$, and it means that *none* of the cells in p_i are blocked.

For ease of notation below, we will use the path name, p_i, to refer both to the path in $\mathcal{P}$ and the set of cells in $\mathcal{C}$ that represents that path, where the meaning will be clear from context. Note that using this representation, $\mathcal{P}$ is a collection of sets of cells.

There are at least two ways to compute $P_{pnb}(\mathcal{P})$. The first simply sums over all possible environments (i.e., configurations of obstacles), checking which ones still possess possible paths:

$$P_{pnb}(\mathcal{P}) = \frac{1}{2^{|\mathcal{C}|}} \sum_{C \in 2^{\mathcal{C}}} [\exists p_i \text{ s.t. } C \cap p_i = \emptyset]. \quad (1)$$

The complexity of this algorithm is $O(mn2^m)$.

The second approach applies the inclusion-exclusion principle to sums over all possible (unions of) subsets of paths, with complexity $O(mn2^n)$:

$$\begin{aligned} P_{pnb}(\mathcal{P}) &= Prob(p_1 \text{ not blocked or } p_2 \text{ not blocked} \\ &\qquad \text{or } \dots p_n \text{ not blocked}) \\ &= \sum_i P_{cnb}(p_i) - \sum_{i \neq j} P_{cnb}(p_i \cup p_j) \\ &\quad + \sum_{i \neq j \neq k} P_{cnb}(p_i \cup p_j \cup p_k) - \cdots \\ &\quad + (-1)^{n-1} P_{cnb}(p_1 \cup p_2 \cup \cdots \cup p_n) \\ &= \sum_{A \in 2^{\mathcal{P}} \setminus \emptyset} (-1)^{|A|-1} P_{cnb}\left(\cup_{p_i \in A} p_i\right), \end{aligned} \quad (2)$$

It only remains to show how to compute $P_{cnb}(A)$, for a singleton *set* of cells A. However, this is given by

$$P_{cnb}(A) = \frac{2^{m-|A|}}{2^m} = \frac{1}{2^{|A|}}, \quad (3)$$

which simply counts the fraction of environments that do not include as obstacles any of the cells in set A.

Using (1), (2) and (3), Problems 1 and 2 can be solved in a brute-force manner, but only efficiently for small m and n. The problems we are interested in have $m \approx 10^5$ and $n \approx 10^4$. Therefore, we must find sub-optimal approximation algorithms to solve these problems.

(a)

Goal	
2	4
1	3
Start	

(b)

Goal			
4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13
Start			

Fig. 2. Environments for (a) Example 1 and (b) Example 2.

III. EXAMPLES

Let's consider a few examples to fix ideas.

Example 1: Consider the 2×2 grid world of Figure 2(a). Consider two paths connecting the Start to the Goal: $p_1 = \{1, 2\}$ and $p_2 = \{3, 4\}$. There are $2^4 = 16$ possible environments, depending on whether cells 1–4 contain an obstacle or not:

$$0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111,$$
$$1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111.$$

Using (1), 7 of these environments (0000, 0001, 0010, 0011, 0100, 1000, 1100) leave at least one of the paths unblocked. So, $P_{pnb}(\mathcal{P}) = 7/16$. Using (2), we compute

$$\begin{aligned} P_{pnb}(\mathcal{P}) &= P_{cnb}(p_1) + P_{cnb}(p_2) - P_{cnb}(p_1 \cup p_2) \\ &= \frac{2^2}{2^4} + \frac{2^2}{2^4} - \frac{2^4}{2^4} = \frac{7}{16}. \end{aligned} \quad (4)$$

Finally, because these paths are independent (i.e., non-overlapping), we can see this another way: there are three ways to block p_1 and three independent ways to block p_2, for a total of $3 \cdot 3 = 9$ ways to block out of 16. Hence, 7 of 16 are not blocked.

Example 2: Consider the 4×4 grid world of Figure 2(b) and the four paths connecting the Start and Goal:

$$p_1 = \{1, 2, 3, 4\}, \qquad p_2 = \{5, 6, 7, 8\},$$
$$p_3 = \{9, 10, 11, 12\}, \qquad p_4 = \{13, 14, 15, 16\}.$$

There are $2^{16} = 65536$ possible environments. Therefore, we will use (2):

$$P_{pnb}(\mathcal{P}) = 4\frac{2^{12}}{2^{16}} - 6\frac{2^8}{2^{16}} + 4\frac{2^4}{2^{16}} - \frac{2^0}{2^{16}} = \frac{14911}{65536}.$$

Again, because these paths are independent (i.e., non-overlapping), we can see this another way: there are 15 ways to block each p_i, for a total of $15^4 = 50625$ ways to block all four. This leaves $65536 - 50625 = 14911$ environments which do not block all four paths.

Example 3: Consider the setup of Example 1, but add the winding path $p_3 = \{1, 3, 4, 2\}$. It is easy to see that this path will not change $P_{pnb}(\mathcal{P})$ using (1), because p_3 has a non-zero intersection (hence is blocked) by all 9 environments that blocked the original set.

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.

Example 4: Consider the same setup as in Example 2, but add the winding path

$$p_5 = \{1, 5, 9, 13, 14, 10, 6, 2, 3, 7, 11, 15, 16, 12, 8, 4\}.$$

Arguing as in Example 3, $P_{pnb}(\mathcal{P})$ will not change using (1). It can also be shown that it will not change using (2). The reason is that p_5 itself (and hence any path set that contains it) contains all 16 cells. Thus, $P_{cnb}(p_5)$ and P_{pnb} for each subset of $\mathcal{P}$ that contains it is equal to $1/2^{16}$. The result then follows from the fact that

$$0 = (1-1)^n = \sum_{i=0}^{n} (-1)^i \begin{pmatrix} n \\ i \end{pmatrix}. \tag{5}$$

IV. APPROXIMATION ALGORITHMS

To approximately solve Problem 2, we propose a greedy algorithm. First, consider mapping each path p_i to an m-digit binary string b_i whose jth digit is 1 if cell c_j is in p_i and 0, otherwise. Thus, for Example 2 above:

$$b_1 = 1111000000000000, \quad b_2 = 0000111100000000,$$
$$b_3 = 0000000011110000, \quad b_4 = 0000000000001111.$$

Define the weight of binary string, $w(b)$, to be the number of 1's in string b. Thus, $w(b_i) = 4$ for each string above. Finally, define component-wise addition of two strings $s \oplus t$. With these, we provide a greedy algorithm for Problem 2, which has complexity $O(mn^2)$:

PATH DIVERSITY INNER-PRODUCT($\mathcal{P}$)
1 $b_{first} \leftarrow \min_{b \in \mathcal{P}} w(b)$
2 $\mathcal{P}' \leftarrow \{b_{first}\}$
3 $s \leftarrow b_{first}$
4 **repeat**
5 $b_{next} \leftarrow \min_{b \in \mathcal{P}}$ dotproduct(b, s)
6 $\mathcal{P} = \mathcal{P} - \{b_{next}\}$
7 $\mathcal{P}' \leftarrow \mathcal{P}' \cup \{b_{next}\}$
8 $s \leftarrow s \oplus b_{next}$
9 **until** $|\mathcal{P}'| = k$

A different greedy algorithm can be formulated using an approximation of (2). When adding a new path p to $\mathcal{P}$, the two most significant terms for the incremental change in $P_{pnb}(\mathcal{P})$ are

$$f(p) = P_{cnb}(p) - \sum_{i=1}^{n} p_{cnb}(p \cup p_i). \tag{6}$$

Consistent with the weight of its binary string, we will define the weight of a path p to be the number of distinct cells it contains. If $w(p) = l$ and $w(p_i) = w_i$, $w(p \cap p_i) = v_i$, for $i = 1, \ldots, n$, then $f(p)$ reduces to

$$f(p) = \frac{1}{2^l} - \sum_{i=1}^{n} \frac{1}{2^{w_i + l - v_i}} = \frac{1}{2^l} \left(1 - \sum_{i=1}^{n} \frac{1}{2^{w_i - v_i}} \right).$$

The idea then, is to pick the p to add at each step that maximizes this expression:

PATH DIVERSITY INCLUSION-EXCLUSION($\mathcal{P}$)
1 $p_{first} \leftarrow \min_{p \in \mathcal{P}} w(p)$
2 $\mathcal{P}' \leftarrow \{p_{first}\}$
3 $s \leftarrow p_{first}$
4 **repeat**
5 $p_{next} \leftarrow \max_{p \in \mathcal{P}} f(p)$
6 $\mathcal{P} = \mathcal{P} - \{p_{next}\}$
7 $\mathcal{P}' \leftarrow \mathcal{P}' \cup \{p_{next}\}$
8 **until** $|\mathcal{P}'| = k$

Both algorithms run in time $O(mn^2)$. For the latter, note that an exact implementation of the powers of two on m bits cannot rely on hardware floating point.

Just as in [14], both of these algorithms exhibit the useful property that they can be terminated at any iteration count to return the n best paths from the set. The key distinction between them is their treatment of space unoccupied by any path in the subset. When choosing a path which occupies previously unreachable cells, the Inner-Product algorithm regards each cell as free (with the exception of the initial path selection). By contrast, the Inclusion-Exclusion algorithm is computing the probability of a collision. Under the assumption that each cell is occupied with probability 0.5, the penalty for extending a path by a single cell is that its survival is half as likely. The contrasting reward that a longer path can accomplish more is excluded from the analysis, resulting in the behavior borne out in the results below.

V. EXPERIMENTAL RESULTS

We applied the approximate path diversity algorithms described above to a large example problem consisting of 34,295 paths of various lengths. These paths were generated from motion capture data for use in graphical human character animation and are the same data used in [4]. The path set was culled down to a variety of sizes using each of the greedy algorithms described above. Since the algorithms require a cellular discretization of space, the world was divided into a rectangular grid of 85 by 70 cells, for a total of 5950 cells. Because we are using a square grid and an irregularly-shaped path set, only 3030 of those cells are traversed by any path. The run time of each of these algorithms was reasonable even for a large problem such as this, in which an exhaustive brute force search would be intractable. The path counts used in this analysis are shown in Table I, along with the run times for each algorithm on each size set.

To aid in visualization, Fig. 1 shows the full path set as well as two example subsets of equal path count representing these two algorithms. For comparison purposes,

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.

Table I. Path set sample sizes and algorithm run times.

Path Count	% of Total Paths	Inner-Product Run Time (sec)	Inclusion-Exclusion Run Time (sec)
343	1	2	260
686	2	3	527
1715	5	7	1438
3430	10	13	3898
6859	20	26	14,189
34,295	100	–	–

a third path sampling algorithm was introduced, which randomly samples paths from the full set. An equal-sized random path set is also shown in Fig. 1. From this figure, it is clear that the path set resulting from the Inner-Product algorithm is more compact than the others, although it contains the same number of paths. This compactness reflects the tendency of the algorithm to select the shortest paths first, since they are most likely to avoid collision with obstacles. This behavior is appropriate so long as all the curves in the original path set are legitimate edges for selection. Some applications may may require that all edges be equal in length, but we leave this issue as future work.

Since these algorithms were designed to maximize the probability that at least one path will not be blocked by any arbitrary obstacle configuration, it is expected that a larger number of paths in the set would survive such obstacles as compared with an uninformed selection of paths. In order to test this speculation, the cells reachable by the full path set were slowly filled with obstacles one-by-one in random order. Each time a new obstacle was found to intersect a path in one of the sets, that path was removed. At each stage, the fraction of paths from each set which survive was plotted. The results of this experiment for a 1% path set, averaged over one thousand trials, are presented in Fig. 3. Here, the vertical axis represents normalized surviving paths. Normalized paths allow one to compare path subset performance against the full path set; the other path set sizes in Table I produce qualitatively similar results.

From these plots, it is clear that the Inner-Product path diversity algorithm consistently outperforms both the full set and the random subset (which closely tracks the former, as would be expected with a uniform sampling). While the Inner-Product algorithm delivered quite good results, the Inclusion-Exclusion algorithm consistently underperforms in these tests. This can be attributed to the fact that (6) – the basis of this algorithm – uses a two-term approximation of the inclusion-exclusion rule. This assumption would give the exact answer in a case where no three paths ever intersect at a single cell. However, in our large dataset, a significant fraction of paths passed through a small number of cells, making (6) an inadequate approximation of $P_{pnb}(\mathcal{P})$.

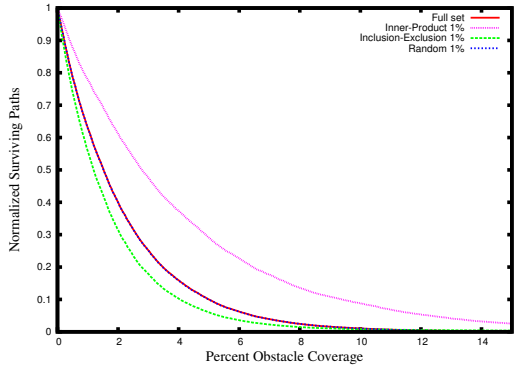

Fig. 3. Normalized survival rate of path sets produced by each algorithm at 1% of the original path set with varying obstacle density. The randomly-sampled subset closely tracks the performance of the full path set, while the Inner-Product algorithm output performs best.

VI. TRAJECTORY DIVERSITY

Formulas and approximation algorithms for trajectory (versus path) diversity directly follow as a corollary of our results in Sections II and IV, by considering the sequence (versus the set) of cells that are traversed. To be specific, the same formulas and algorithms as above may be applied to trajectories by merely expanding the configuration space one dimension to include (discretized) time and then computing/approximating path diversity on the resulting paths in space-time.

The result would be formulas and approximation algorithms that maximize *spatio-temporal path diversity*, or briefly *trajectory diversity*. These would favor pruning of overlapping trajectories and the resulting trajectory sets would be those that have the lowest probability of being blocked, averaged uniformly over all space-time obstacle environments.

The benefit of computing/approximating tree prunings that maximize such spatio-temporal diversity would be lost unless one considers environments with dynamic obstacles. The reason is that, in the completely static case, the sum over all environments in (1) should not treat obstacles as arbitrary in both space and time. Instead, such obstacles would be "columns" in space time: they would have arbitrary bases in space, but extend across the complete time set.

It is future work to determine whether space and time should be have equal weighting when computing (1) or its approximations herein, even in the case of dynamic obstacles. Should different weightings be used for different bounded obstacle velocities?

Finally, it is interesting to note that even though paths may overlap greatly in space, they may have many fewer intersections as trajectories in space-time.

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.

VII. SUMMARY AND DISCUSSION

The eventual goal of this work is to aid in the design of precomputed path sets. Through principled path selection, planning efficiency can be maximized and the probability of failure minimized. In the process of evaluating a path set against a collection of obstacles, the collision of one path provides information which can be used to inform future path selection. In an uninformed path set, computation will be, in essence, wasted by repeatedly colliding different paths with the same obstacles. A planner could use this information explicitly by generating a decision tree of paths, but a static path set may also elicit the desired behavior. Our algorithms decrease this wasted computation by selecting a path set that minimizes the amount of overlap between paths at all points in the world.

As we have shown, when planning with such paths, the survival rate is maximized, so fewer cycles are wasted on paths that collide. With more viable options produced per path evaluation, the likelihood of the goal being achieved is maximized. Due to extreme combinatorial complexity, it is impractical to compute the exact probability of collision of a particular path set. However, we have shown that approximate algorithms can be effectively employed to design a diverse path set that is expected to perform well during planning.

This work represents progress toward understanding how best to prune existing sets of paths or reachability sets as well as providing information to help make informed decisions about trajectory set design. The presented algorithms work with any path set, including those generated incrementally (say, by an RRT), or by a model that incorporates dynamic constraints. Because our methods work on resulting paths/trajectories, they are also agnostic with respect to whether the generated paths are parametrized or non-parametrized. In the former case, it may be possible to obtain diversity in specific cases. However, if the dynamics are non-trivial (non-linear, non-holonomic, etc.), it may be expensive or impossible to generate diverse paths through judicious parameter selection. Also in such general cases, RRTs produce non-separated—even overlapping—workspace paths. Algorithms like ours that work on resulting paths offer a general solution to the diversity problem.

We have explored discretized versions of the path diversity problem using numerical simulations. One avenue of future work would involve designing analytical metrics for path diversity along the lines of [14] to eliminate arbitrary discretizations from the analysis. Another direction for future work is to perform experiments on space-time trajectories with dynamic obstacles, for which the Inclusion-Exclusion algorithm may be a better match. Finally, yet another direction is to experiment with non-uniform obstacle distributions and to extend the framework and algorithms for pruning to environments where one has only partial obstacle information (say, partial occupancy from LIDAR or vision sensing) but also has a statistical description of the environment in terms of a *Markov random field* (MRF). Such MRF descriptions would have joint (versus independent) probabilities of obstacles occurring in a given cell, based on the presence of obstacles in its neighboring cells. We anticipate that such models would be especially useful in terms of path diversity and pruning for the types of natural environments mentioned in the Introduction and also in virtual natural environments where such fields are used to "texture," say, forests or other terrain, planet surfaces, or even city environments. Similar arguments apply to dynamic, multi-agent environments where occupancy probabilities would come from traffic models and/or Kalman filter biases arising from vehicle tracking.

Acknowledgments: This work was partially supported by NSF grants CCR-0208919 (Branicky) and EEC-0540865 (Kuffner) and DARPA (Knepper), The authors gratefully acknowledge Manfred Lau for making the large motion capture path set available.

REFERENCES

[1] E. Frazzoli, M. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002.

[2] P. Leven and S. Hutchinson, "A framework for real-time path planning in changing environments," *Intl. J. Robotics Research*, vol. 21, no. 12, p. 999, 2002.

[3] J. Go, T. Vu, and J. Kuffner, "Autonomous behaviors for interactive vehicle animations," *Int. J. Graphical Models*, 2005.

[4] M. Lau and J. Kuffner, "Precomputed search trees: Planning for interactive goal-driven animation," in *Proc. ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2006.

[5] S. Thrun *et al.*, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[6] L. Kavraki *et al.* "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] S. LaValle and J. Kuffner, "Randomized Kinodynamic Planning," *The Intl. J. Robotics Research*, vol. 20, no. 5, p. 378, 2001.

[8] R. Bohlin and L. Kavraki, "A lazy probabilistic roadmap planner for single query path planning," *Intl. J. Robotics Research*, 2000.

[9] M. Branicky *et al.*, "Quasi-randomized path planning," *Proc. IEEE Intl. Conf. Robotics and Automation*, vol. 2, 2001.

[10] B. Burns and O. Brock, "Toward optimal configuration space sampling," *Proc. Robotics: Science and Systems*.

[11] S. Ramamoorthy *et al.*, "Low-discrepancy curves and efficient coverage of space," *Alg. Foundations of Robotics VII*, 2006.

[12] A. Bicchi *et al.*, "On the reachability of quantized control systems," *IEEE Trans. Automatic Control*, 47(4):546–563, 2002.

[13] S. Goldberg, M. Maimone, and L. Matthies, "Stereo Vision and Rover Navigation Software for Planetary Exploration," *Proceedings of the 2002 IEEE Aerospace Conference*, pp. 2025–36, 2002.

[14] C. Green and A. Kelly, "Toward optimal sampling in the space of paths," in *Proc. Intl. Symp. Robotic Research*, Hiroshima, 2007.

Authorized licensed use limited to: Technische Hochschule Ingolstadt. Downloaded on October 06,2022 at 06:58:06 UTC from IEEE Xplore. Restrictions apply.