An Equivalence Relation for Local Path Sets

Ross A. Knepper, Siddhartha S. Srinivasa, and Matthew T. Mason

Abstract We propose a novel enhancement to the task of collision-testing a set of local paths. Our approach circumvents expensive collision-tests, yet it declares a continuum of paths collision-free by exploiting both the structure of paths and the outcome of previous tests. We define a homotopy-like equivalence relation among local paths and provide algorithms to (1) classify paths based on equivalence, and (2) implicitly collision-test up to 90% of them. We then prove both correctness and completeness of these algorithms before providing experimental results showing a performance increase up to 300%.

1 Introduction

Planning bounded-curvature paths for mobile robots is an NP-hard problem [22]. Many nonholonomic mobile robots thus rely on hierarchical planning architectures [1, 13, 19], which split responsibility between at least two layers (Fig. 1): a slow global planner and fast local planner. We focus here on the local planner (Alg. 1 and Alg. 2), which iterates in a tight loop: searching through a set of paths and selecting the best path for execution. During each loop, the planner tests many paths before making an informed decision. The bottleneck in path testing is collision-testing [24]. In this paper, we introduce a novel approach that delivers a significant increase in path set collision-testing performance by exploiting the fundamental geometric structure of paths.

We introduce an equivalence relation intuitively resembling the topological notion of homotopy. Two paths are *path homotopic* if a continuous, collision-free deformation with fixed start and end points exists between them [20]. Like any path

Ross A. Knepper and Matthew T. Mason
Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh PA, e-mail: {rak,mason}@ri.cmu.edu

Siddhartha S. Srinivasa
Intel Labs Pittsburgh, 4720 Forbes Ave, Pittsburgh, PA e-mail: siddhartha.srinivasa@intel.com

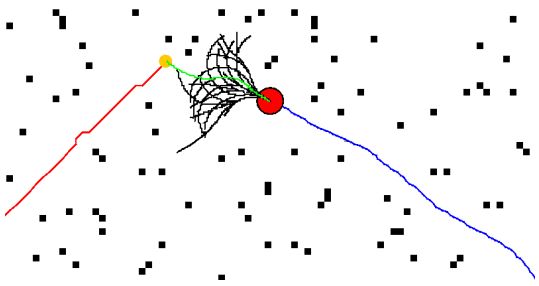


Fig. 1 An example hierarchical planning scenario. The local planner's path set expands from the robot, at center, and feeds commands to the robot based on the best path that avoids obstacles (black squares). The chosen local path (green) and global path (red) combine to form a proposed path to the goal.

equivalence relation, homotopy partitions paths into equivalence classes. Different homotopy classes make fundamentally different choices about their route amongst obstacles. However, two mobile robot concepts translate poorly into homotopy theory: limited sensing and constrained action.

The robot may lack a complete workspace map, which must instead be constructed from sensor data. Since robot perception is limited by range and occlusion, a robot's understanding of obstacles blocking its movement evolves with its vantage point. A variety of sensor-based planning algorithms have been developed to handle such partial information. Obstacle avoidance methods, such as potential fields [12], are purely reactive. The bug algorithm [18], which generates a path to the goal using only a contact sensor, is complete in 2D. Choset and Burdick [5] present the hierarchical generalized Voronoi graph, a roadmap with global line-of-sight accessibility that achieves completeness in higher dimensions using range readings of the environment.

If a robot is tasked to perform long-range navigation, then it must plan a path through unsensed regions. A low-fidelity global planner generates this path because we prefer to avoid significant investment in this plan, which will likely be invalidated later. Path homotopy, in the strictest sense, requires global knowledge of obstacles because homotopy equivalent paths must connect fixed start and goal points.

Relaxing the endpoint requirement avoids reasoning about the existence of faraway, unsensed obstacles. Naively relaxing a fixed endpoint, our paths might be permitted to freely deform around obstacles, making all paths equivalent. To restore meaningful equivalence classes, we propose an alternate constraint based on path shape. This is in keeping with the nonholonomic constraints that limit mobile robots' action. Laumond [15] first highlighted the importance of nonholonomic constraints and showed that feasible paths exist for a mobile robot with such constraints. Barraquand and Latombe [2] created a grid-based planner that innately captures these constraints. LaValle and Kuffner [17] proposed the first planner to incorporate both kinodynamic constraints and random sampling. In contrast to nonholonomic constraints, true homotopy forbids restrictions on path shape; two paths are equivalent if any path deformation—however baroque—exists between them. By restricting our paths to bounded curvature, we represent only feasible motions while limiting paths' ability to deform around obstacles. The resulting set of path equivalence classes is of immediate importance to the planner (Fig. 2). The number

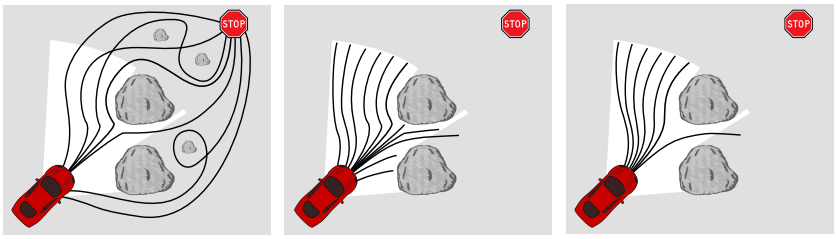


Fig. 2 left: Paths from a few distinct homotopy classes between the robot and the goal. The distinctions between some classes require information that the robot has not yet sensed (the dark area is out of range or occluded). **middle:** With paths restricted to the sensed area, they may freely deform around visible obstacles. **right:** After restricting path shape to conform to motion constraints, we get a handful of equivalence classes that are immediately applicable to the robot.

of choices represented by these local equivalence classes relates to Farber's topological complexity of motion planning [6].

Equivalence classes have been employed in various planners. In task planning, recent work has shown that equivalence classes of actions can be used to eliminate redundant search [7]. In motion planning, path equivalence often employs homotopy. A recent paper by Bhattacharya, Kumar, and Likhachev [3] provides a technique based on complex analysis for detecting homotopic equivalence among paths in 2D. Two papers employing equivalence classes to build probabilistic roadmaps [11] are by Schmitzberger, et al. [25] and Jaillet and Simeon [10]. The latter paper departs from true homotopy by proposing the visibility deformation, a simplified alternative to homotopic equivalence based on line-of-sight visibility between paths.

Our key insight is that local path equivalence is an expressive and powerful tool that reveals shared outcomes in collision-testing. Specifically, two equivalent neighboring paths cover some common ground in the workspace, and between them lies a continuum of covered paths. We develop the mathematical foundations to detect equivalence relations among all local paths based on a finite precomputed path set. We then utilize these tools to devise efficient algorithms for detecting equivalence and implicitly collision-testing local paths.

The remainder of the paper is organized as follows. We provide an implementation of the basic algorithm in Section 2 and present the fast collision-testing technique. Section 3 then explores the theoretical foundations of our path equivalence relation. Section 4 provides some experimental results.

2 Algorithms

In this section, we present three algorithms: path set generation, path classification, and implicit path collision-testing. All of the algorithms presented here run in polynomial time. Throughout this paper, we use lowercase p to refer to a path in the workspace, while $\mathcal{P}$ is a set of paths (each one a point in path space).

Algorithm 1 Test_All_Paths(w, $\mathcal{P}$)

Input: w – a costmap object; $\mathcal{P}$ – a fixed set of paths
Output: $\mathcal{P}_{free}$, the set of paths that passed collision test
1: $\mathcal{P}_{free} \leftarrow \emptyset$
2: **while** time not expired **and** untested paths remain **do** // test paths for 0.1 seconds
3: $p \leftarrow Get_Next_Path(\mathcal{P})$
4: $collision \leftarrow w.Test_Path(p)$ // *collision* is boolean
5: **if** not *collision* **then**
6: $\mathcal{P}_{free} \leftarrow \mathcal{P}_{free} \cup \{p\}$ // non-colliding path set
7: **return** $\mathcal{P}_{free}$

Algorithm 2 Local_Planner_Algorithm(w, x, h, $\mathcal{P}$)

Input: w – a costmap object; x – initial state; h – a heuristic function for selecting a path to execute; $\mathcal{P}$ – a fixed set of paths
Output: Moves the robot to the goal if possible
1: **while** not at goal **and** time not expired **do**
2: $\mathcal{P}_{free} \leftarrow Test_All_Paths(w, x, \mathcal{P})$
3: $j \leftarrow h.Best_Path(x, \mathcal{P}_{free})$
4: $Execute_Path_On_Robot(j)$
5: $x \leftarrow Predict_Next_State(x, j)$

Definition 1. *Path space* is a metric space $(\mathcal{P}, \mu)$ in which the distance between a pair of paths in $\mathcal{P}$ is defined by metric μ. Paths can vary in shape and length. □

2.1 Path Set Generation

We use the greedy path set construction technique of Green and Kelly [8], outlined in Alg. 3. The algorithm iteratively builds a path set $\mathcal{P}_N$ by drawing paths from a densely-sampled source path set, $\mathcal{X}$. During step i, it selects the path $p \in \mathcal{X}$ that minimizes the dispersion of $\mathcal{P}_i = \mathcal{P}_{i-1} \cup \{p\}$. Borrowing from Niederreiter [21]:

Definition 2. Given a bounded metric space $(\mathcal{X}, \mu)$ and point set $\mathcal{P} = \{x_1, \ldots, x_N\} \in \mathcal{X}$, the *dispersion* of $\mathcal{P}$ in $\mathcal{X}$ is defined by

$$\delta(\mathcal{P}, \mathcal{X}) = \sup_{x \in \mathcal{X}} \min_{p \in \mathcal{P}} \mu(x, p). \quad \square \tag{1}$$

The dispersion of $\mathcal{P}$ in $\mathcal{X}$ equals the radius of the biggest open ball in $\mathcal{X}$ containing no points in $\mathcal{P}$. By minimizing dispersion, we ensure that there are no large voids in path space. Thus, dispersion reveals the quality of $\mathcal{P}$ as an "approximation" of $\mathcal{X}$ because it guarantees that for any $x \in \mathcal{X}$, there is some point $p \in \mathcal{P}$ such that $\mu(x, p) \leq \delta(\mathcal{P}, \mathcal{X})$.

The Green-Kelly algorithm generates a sequence of path sets $\mathcal{P}_i$, for $i \in \{1, \ldots, N\}$, that has monotonically decreasing dispersion. Alg. 1 searches paths in this order at runtime, thus permitting early termination while retaining near-optimal results. Note

that while the source set $\mathcal{X}$ is of finite size—providing a lower bound on dispersion at runtime—it can be chosen with arbitrarily low dispersion a priori.

Algorithm 3 Green_Kelly($\mathcal{X}$, N)

Input: $\mathcal{X}$ – a densely-sampled, low-dispersion path set; $N \leq |\mathcal{X}|$ – the target path set size
Output: path sequence $\mathcal{P}_N$ of size N

$\mathcal{P}_0 \leftarrow \emptyset$
$n \leftarrow 0$
while $n < N$ **do**
    $n \leftarrow n+1$
    $p \leftarrow \underset{x \in \mathcal{X}}{\operatorname{argmin}}\ \delta(\mathcal{P}_{n-1} \cup \{x\}, \mathcal{X})$
    $\mathcal{P}_n \leftarrow \mathcal{P}_{n-1} \cup \{p\}$
return $\mathcal{P}_N$

2.2 Path Classification

We next present Alg. 4, which classifies collision-free members of a path set. The Hausdorff metric is central to the algorithm. Intuitively, this metric returns the greatest amount of separation between two paths in the workspace. From Munkres [20]:

$$\mu_H(p_i, p_j) = \inf\{\varepsilon \colon p_i \subset (p_j)_\varepsilon \text{ and } p_j \subset (p_i)_\varepsilon\}, \tag{2}$$

where $(p)_r$ denotes dilation of p by r: $\{t \in \mathbb{R}^2 : \|t_p - t\|_{L2} \leq r \text{ for some } t_p \in p\}$. Note that μ_H satisfies all properties of a metric [9]. For our fixed path set generated by Green-Kelly, we precomputed each pairwise path metric value of (2) and stored them in a lookup table for rapid online access.

Alg. 4 performs path classification on a set of paths that have already tested collision-free at runtime. We form a graph $G = (V, E)$ in which node $v_i \in V$ corresponds to path p_i. Edge $e_{ij} \in E$ joins nodes v_i and v_j when this relation holds:

$$\mu_H(p_i, p_j) \leq d, \tag{3}$$

where d is the diameter of the robot. Taking the transitive closure of this relation, two paths p_a and p_b are equivalent if nodes v_a and v_b are in the same connected component of G (Fig. 3).

In effect, this algorithm constructs a probabilistic roadmap (PRM) in the path space instead of the conventional configuration space. A query into this PRM tells whether two paths are equivalent. As with any PRM, a query is performed by adding two new graph nodes v_s and v_g corresponding to the two paths. We attempt to join these nodes to other nodes in the graph based on (3). The existence of a path connecting v_s to v_g indicates path equivalence.

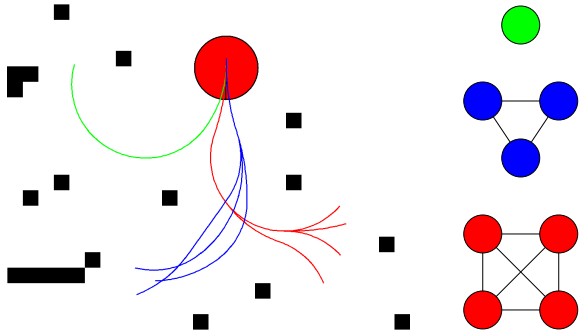


Fig. 3 A simple path set, in which obstacles (black) eliminate colliding paths. The collision-free path set has three equivalence classes (red, green, and blue). In the corresponding graph representation, at right, adjacent nodes represent proximal paths. Connected components indicate equivalence classes of paths.

Algorithm 4 Equivalence_Classes($\mathcal{P}_{free}$, d)

Input: $\mathcal{P}_{free}$ – a set of safe, appropriate paths; d – the diameter of the robot
Output: D – a partition of $\mathcal{P}_{free}$ into equivalence classes (a set of path sets)

Let $G = (V, E) \leftarrow (\emptyset, \emptyset)$
$D \leftarrow \emptyset$
for all $p_i \in \mathcal{P}_{free}$ **do** // This loop discovers adjacency
    $V.add(p_i)$ // Add a graph node corresponding to path p_i
    for all $p_j \in V \setminus \{p_i\}$ **do**
        if $\mu_H(p_i, p_j) < d$ **then**
            $E.add(i, j)$ // Connect nodes i and j with an unweighted edge
$\mathcal{S} \leftarrow \mathcal{P}_{free}$
while $\mathcal{S} \neq \emptyset$ **do** // This loop finds the connected components
    $\mathcal{C} \leftarrow \emptyset$
    $p \leftarrow$ a member of $\mathcal{S}$
    $\mathcal{L} \leftarrow \{p\}$ // List of nodes to be expanded in this class
    while $\mathcal{L} \neq \emptyset$ **do**
        $p \leftarrow$ a member of $\mathcal{L}$
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{p\}$ // Commit p to class
        $\mathcal{S} \leftarrow \mathcal{S} \setminus \{p\}$
        $\mathcal{L} \leftarrow (\mathcal{L} \cup V.neighbors(p)) \cap \mathcal{S}$
    $D \leftarrow D \cup \{\mathcal{C}\}$
return D

2.3 Implicit Path Safety Test

There is an incessant need in motion planning to accelerate collision-testing, which may take 99% of total CPU time [24]. During collision-testing, the planner must verify that a given swath is free of obstacles.

Definition 3. A *swath* is the workspace area of ground or volume of space swept out as the robot traverses a path. □

Definition 4. We say a path is *safe* if its swath contains no obstacles. □

In testing many swaths of a robot passing through space, most planners effectively test the free workspace many times by testing overlapping swaths. We may test a path implicitly at significant computational savings by recalling recent

collision-testing outcomes. We formalize the idea in Alg. 5, which is designed to be invoked from Alg. 1, line 4 in lieu of the standard path test routine.

The implicit collision-test condition resembles the neighbor condition (3) used by Alg. 4, but it has an additional "Is_Between" check, which indicates that the swath of the path under test is covered by two collision-free neighboring swaths. The betweenness trait can be precomputed and stored in a lookup table. Given a set of safe paths, we can quickly discover whether any pair covers the path under test. Experimental results show that this algorithm allows us to test up to 90% of paths implicitly, thus increasing the path evaluation rate by up to 300% in experiments.

Algorithm 5 Test_Path_Implicit(p, w, $\mathcal{S}$, d)

Input: p is a path to be tested
Input: w is a costmap object // used as a backup when path cannot be implicitly tested
Input: $\mathcal{S}$ is the set of safe paths found so far
Input: d is the diameter of the robot
1: **for all** $p_i, p_j \in \mathcal{S}$ such that $\mu_H(p_i, p_j) \leq d$ **do**
2: **if** $p.Is_Between(p_i, p_j)$ **then** // p's swath has been tested previously
3: $s_f \leftarrow p.Get_End_Point()$
4: $collision \leftarrow w.Test_Point(s_f)$ // endpoint may not be covered by swaths
5: **return** $collision$
6: **return** $w.Test_Path(p)$ // Fall back to explicit path test

3 Foundations

In this section, we establish the foundations of an equivalence relation on path space based on continuous deformations between paths. We then provide correctness proofs for our algorithms for classification and implicit collision-testing.

We assume a kinematic description of paths. All paths are parametrized by a shared initial pose, shared fixed length, and individual curvature function. Let $\kappa_i(s)$ describe the curvature control of path i as a function of arc length, with $\max_{0\leq s\leq s_f} |\kappa_i(s)| \leq \kappa_{max}$. Typical expressions for κ_i include polynomials, piecewise constant functions, and piecewise linear functions. The robot motion produced by control i is a *feasible* path given by

$$\begin{bmatrix} \dot{\theta}_i \\ \dot{x}_i \\ \dot{y}_i \end{bmatrix} = \begin{bmatrix} \kappa_i \\ \cos\theta_i \\ \sin\theta_i \end{bmatrix}. \tag{4}$$

Definition 5. A *feasible* path has bounded curvature (implying C^1 continuity) and fixed length. The set $\mathcal{F}(s_f, \kappa_{max})$ contains all feasible paths of length s_f and curvature $|\kappa(s)| \leq \kappa_{max}$. □

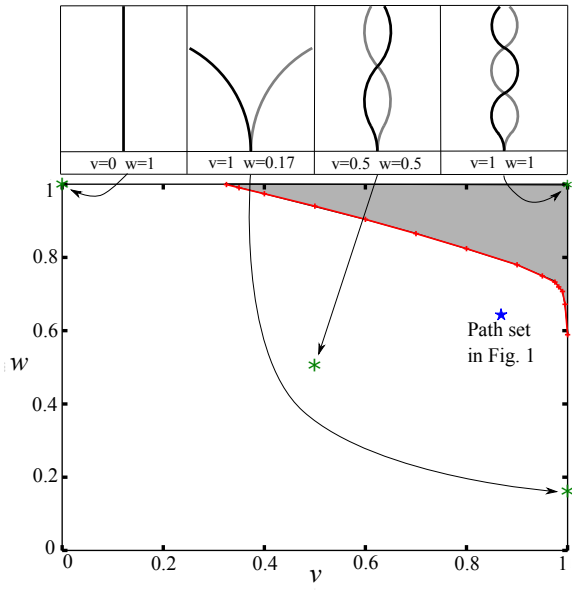


Fig. 4 At top: several example paths combining different values of v and w. Each path pair obeys (3). The value of v affects the "curviness" allowed in paths, while w affects their length. **At bottom:** this plot, generated numerically, approximates the set of appropriate choices for v and w. The gray region at top right must be avoided, as we show in Lemma 2. Such choices would permit an obstacle to occur between two safe paths that obey (3). A path whose values fall in the white region is called an *appropriate path*.

3.1 Properties of Paths

In this section, we establish a small set of conditions under which we can quickly determine that two paths are equivalent. We constrain path shape through two dimensionless ratios relating three physical parameters. We may then detect equivalence through a simple test on pairs of paths using the Hausdorff metric.

These constraints ensure a continuous deformation between neighboring paths while permitting a range of useful actions. Many important classes of action sets obey these general constraints, including the line segments common in RRT [17] and PRM planners, as well as constant curvature arcs. Fig. 1 illustrates a more expressive action set [13] that adheres to our constraints.

The three physical parameters are: d, the diameter of the robot; s_f, the length of each path; and r_{min}, the minimum radius of curvature allowed for any path. Note that $1/r_{min} = \kappa_{max}$, the upper bound on curvature. For non-circular robots, d reflects the minimal cross-section of the robot's swath sweeping along a path. We express relationships among the three physical quantities by two dimensionless parameters:

$$v = \frac{d}{r_{min}} \qquad\qquad w = \frac{s_f}{2\pi r_{min}}.$$

We only compare paths with like values of v and w. Fig. 4(top) provides some intuition on the effect of these parameters on path shape. Due to the geometry of paths, only certain choices of v and w are appropriate.

Definition 6. An *appropriate path* is a feasible path conforming to appropriate values of v and w from the proof of Lemma 2. Fig. 4 previews the permissible values.

When the condition in (3) is met, the two paths' swaths overlap, resulting in a continuum of coverage between the paths. This coverage, in turn, ensures the existence of a continuous deformation, as we show in Theorem 1, but first we formally define a continuous deformation between paths.

Definition 7. A *continuous deformation* between two safe, feasible paths p_i and p_j in $\mathcal{F}(s_f, \kappa_{max})$ is a continuous function $f\colon [0,1] \to \mathcal{F}(s_f^-, \kappa_{max}^+)$, with s_f^- slightly less than s_f and κ_{max}^+ slightly more than κ_{max}. $f(0)$ is the initial interval of p_i, and $f(1)$ is the initial interval of p_j, both of length s_f^-. We write $p_i \sim p_j$ to indicate that a continuous deformation exists between paths p_i and p_j, and they are therefore equivalent. □

The length s_f^- depends on v and w, but for typical values, s_f^- is fully 95–98% of s_f. For many applications, this is sufficient, but an application can quickly test the remaining path length if necessary. Nearly all paths $f(c)$ are bounded by curvature κ_{max}, but it will turn out that in certain geometric circumstances, the maximum curvature through a continuous deformation is up to $\kappa_{max}^+ = \frac{4}{3}\kappa_{max}$.

Definition 8. Two safe, feasible paths that define a continuous deformation are called *guard paths* because they protect the intermediate paths. □

In the presence of obstacles, it is not trivial to determine whether a continuous deformation is safe, thus maintaining equivalency. Rather than trying to find a deformation between arbitrary paths, we propose a particular condition under which we show that a bounded-curvature, fixed-length, continuous path deformation exists,

$$\mu_H(p_1, p_2) \leq d \implies p_1 \sim p_2. \tag{5}$$

This statement, which we prove in the next section, is the basis for Alg. 4 and Alg. 5. The overlapping swaths of appropriate paths p_1 and p_2 cover a continuum of intermediate swaths between the two paths. Eqn. (5) is a proper equivalence relation because it possesses each of three properties:

- **reflexivity**. $\mu_H(p,p) = 0$; p is trivially deformable to itself.
- **symmetry**. The Hausdorff metric is symmetric.
- **transitivity**. Given $\mu_H(p_1, p_2) \leq d$ and $\mu_H(p_2, p_3) \leq d$, a continuous deformation from p_1 to p_3 passes through p_2.

3.2 Equivalence Relation

Having presented the set of conditions under which (5) holds, we now prove that they are sufficient to ensure the existence of a continuous deformation between two

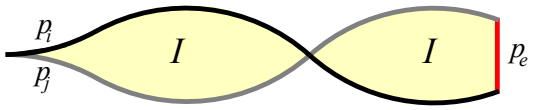


Fig. 5 Paths p_i, p_j, and p_e form boundary B. Its interior, I, contains all paths in the continuous deformation from p_i to p_j.

neighboring paths. Our approach to the proof will be to first describe a feasible continuous deformation, then show that paths along this deformation are safe.

Given appropriate guard paths p_i and p_j with common origin, let p_e be the shortest curve in the workspace connecting their endpoints without crossing either path (p_e may pass through obstacles). The closed path $B = p_i \cup p_j \cup p_e$ creates one or more closed loops (the paths may cross each other). By the Jordan curve theorem, each loop partitions $\mathbb{R}^2$ into two sets, only one of which is compact. Let I, the interior, be the union of these compact regions with B, as in Fig. 5.

Definition 9. A path p_c is *between* paths p_i and p_j if $p_c \subset I$. □

Lemma 1. *Given appropriate paths $p_i, p_j \subset \mathcal{F}(s_f, \kappa_{max})$ with $\mu_H(p_i, p_j) \leq d$, a path sequence exists in the form of a feasible continuous deformation between p_i and p_j.*

Proof. We provide the form of a continuous deformation from p_i to p_j such that each intermediate path is between them. With t a workspace point and p a path, let

$$\gamma(t,p) = \inf\{\varepsilon : t \in (p)_\varepsilon\} \tag{6}$$

$$g(t) = \begin{cases} [0,1] & \text{if } \gamma(t,p_i) = \gamma(t,p_j) = 0 \\ \left\{ \frac{\gamma(t,p_i)}{\gamma(t,p_i)+\gamma(t,p_j)} \right\} & \text{otherwise,} \end{cases} \tag{7}$$

where $g(t)$ is a set-valued function to accommodate intersecting paths. Each level set $g(t) = c$ for $c \in [0,1]$ defines a weighted generalized Voronoi diagram (GVD) forming a path as in Fig. 6. We give the form of a continuous deformation using level sets $g^{-1}(c)$; each path is parametrized starting at the origin and extending for a length s_f^- in the direction of p_e. Let us now pin down the value of s_f^-. Every point t_i on p_i forms a line segment projecting it to its nearest neighbor t_j on p_j (and vice versa). Their collective area is shown in Fig. 7. Eqn. (3) bounds each segment's length at d. s_f^- is the greatest value such that no intermediate path of length s_f^- departs from the region covered by these projections.

For general shapes in $\mathbb{R}^2$, the GVD forms a set of curves meeting at branching points [23]. In this case, no GVD cusps or branching points occur in any intermediate path. Since $d < r_{min}$, no center of curvature along either guard path can fall in I [4]. Therefore, each level set defines a path through the origin.

Each path's curvature function is piecewise continuous and everywhere bounded. A small neighborhood of either guard path approximates constant curvature. A GVD curve generated by two constant-curvature sets forms a conic section [27]. Table 1 reflects that the curvature of p_c is everywhere bounded with the maximum possible curvature being bounded by $\frac{4}{3}\kappa_{max}$. For the full proofs, see [14]. □

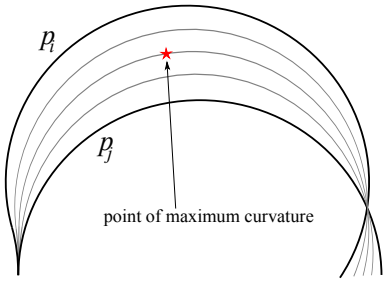


Fig. 6 In a continuous deformation between paths p_i and p_j, as defined by the level sets of (7), each path takes the form of a weighted GVD. Upper bounds on curvature vary along the deformation, with the maximum bound of $\frac{4}{3}\kappa_{max}$ occurring at the medial axis of the two paths.

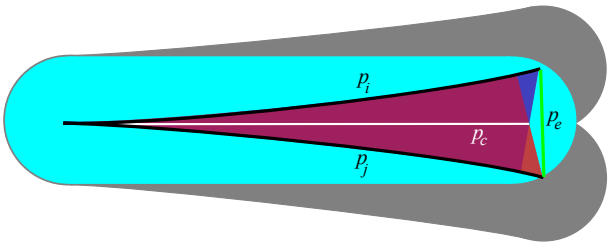


Fig. 7 Hausdorff coverage (overlapping red and blue shapes in center) is a conservative approximation of swath coverage (gray). The Hausdorff distance between paths p_i and p_j is equal to the maximum-length projection from any point on either path to the closest point on the opposite path. Each projection implies a line segment. The set of projections from the top line (blue) and bottom line (red) each cover a solid region between the paths. These areas, in turn, cover a slightly shorter intermediate path p_c, in white, with its swath in cyan. This path's length, s_f^- is as great as possible while remaining safe, with its swath inside the gray area.

Table 1 Conic sections form the weighted Voronoi diagram. κ_1 and κ_2 represent the curvatures of the two guard paths, with κ_1 the lesser magnitude. Let $\kappa_m = \max(|\kappa_1|, |\kappa_2|)$. For details, see [14].

Type	Occurrence	Curvature bounds of intermediate paths
line	$\kappa_1 = -\kappa_2$	$\lvert\kappa\rvert \leq \kappa_m$
parabola	$\kappa_1 = 0, \kappa_2 \neq 0$	$\lvert\kappa\rvert \leq \kappa_m$
hyperbola	$\kappa_1\kappa_2 < 0, \kappa_1 \neq -\kappa_2$	$\lvert\kappa\rvert \leq \kappa_m$
ellipse	$\kappa_1\kappa_2 > 0$	$\lvert\kappa\rvert < \frac{4}{3}\kappa_m$

Lemma 2. *Given safe, appropriate guard paths $p_i, p_j \in \mathcal{F}(s_f, \kappa_{max})$ separated by $\mu_H(p_i, p_j) \leq d$, any path $p_c \subset \mathcal{F}(s_f^-, \frac{4}{3}\kappa_{max})$ between them is safe.*

Proof. We prove this lemma by contradiction. Assume an obstacle lies between p_i and p_j. We show that this assumption imposes lower bounds on v and w. We then conclude that for lesser values of v and w, no such obstacle can exist.

Let $sl(p,d) = \{t \in \mathbb{R}^2, t_p = nn(t,p) : \overline{t_p t} \perp p \text{ and } \|t - t_p\|_{L2} \leq \frac{d}{2}\}$ define a conservative approximation of a swath, obtained by sweeping a line segment of length d with its center along the path. $\overline{t_p t}$ is the line segment joining t_p to t and $nn(t,p)$ is the nearest neighbor of point t on path p. The two swaths form a safe region, $U = sl(p_i,d) \cup sl(p_j,d)$.

Suppose that U contains a hole, denoted by the set h, which might contain an obstacle. Now, consider the shape of the paths that could produce such a hole. Be-

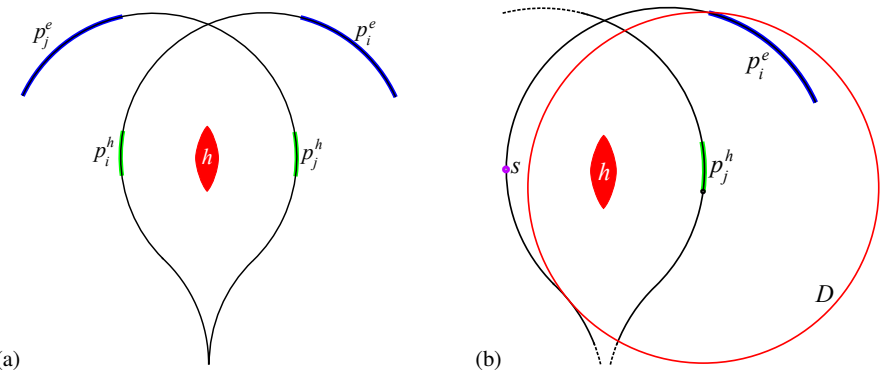


Fig. 8 (a) With bounded curvature, there is a lower bound on path lengths that permit a hole, h, while satisfying (3). Shorter path lengths ensure the existence of a safe continuous deformation between paths. (b) We compute the maximal path length that prevents a hole using Vendittelli's solution to the shortest path for a Dubins car. Starting from the dot marked s, we find the shortest path intersecting the circle D. The interval p_i^e illustrates path lengths permitting a hole to exist.

ginning with equal position and heading, they must diverge widely enough to separate by more than d. To close the loop in U, the paths must then bend back towards each other. Since the paths separate by more than d, there exist two open intervals $p_i^h \subset p_i$ and $p_j^h \subset p_j$ surrounding the hole on each path such that (at this point) $p_i^h \not\subset (p_j)_d$ and $p_j^h \not\subset (p_i)_d$. To satisfy (3), there must exist later intervals $p_i^e \subset p_i$ such that $p_j^h \subset (p_i^e)_d$ and likewise $p_j^e \subset p_j$ such that $p_i^h \subset (p_j^e)_d$, as in Fig. 8a.

How long must a path be to satisfy this condition? Consider the minimum length solution to this problem under bounded curvature. For each point $t \in p_j^h$, the interval p_i^e must intersect the open disc $D = \text{int}((t)_d)$, as in Fig. 8b. Since p_j^h grows with the width of h, and p_i^e must intersect all of these open neighborhoods, the path becomes longer with larger holes. We will therefore consider the minimal small-hole case.

Vendittelli [26] solves the shortest path problem for a Dubins car to reach a line segment. We may approximate the circular boundary of D by a set of arbitrarily small line segments. One may show from this work that given the position and slope of points along any such circle, the shortest path to reach its boundary (and thus its interior) is a constant-curvature arc of radius r_{min}. In the limit, as v approaches one and the size of h approaches zero, the length of arc needed to satisfy (3) approaches $\pi/2$ from above, resulting in the condition that $w > 0.48$. Thus, for $w \leq 0.48$ and $v \in [0,1)$, p_c is safe. For smaller values of v, D shrinks relative to r_{min}, requiring longer paths to reach, thus allowing larger values of w as shown in the plot in Fig. 4.

We have shown that there exist appropriate choices for v and w such that (3) implies that U contains no holes. Since U contains the origin, any path $p_c \in I$ emanating from the origin passes through U and is safe. □

Theorem 1. *Given safe, appropriate guard paths $p_i, p_j \in \mathcal{F}(s_f, \kappa_{max})$, and given $\mu_H(p_i, p_j) \leq d$, a safe continuous deformation exists between p_i and p_j.*

Proof. Lemma 1 shows that (7) gives a continuous deformation between paths p_i and p_j such that each intermediate path $p_c \subset I$ is feasible. Lemma 2 shows that any such path is safe. Therefore, a continuous deformation exists between p_i and p_j. This proves the validity of the Hausdorff metric as a test for path equivalence. □

3.3 Resolution Completeness of Path Classifier

In this section, we show that Alg. 4 is resolution complete. Resolution completeness commonly shows that for a sufficiently high discretization of each dimension of the search space, the planner finds a path exactly when one exists in the continuum space. We instead show that for a sufficiently low dispersion in the infinite-dimensional path space, the approximation given by Alg. 4 has the same connectivity as the continuum safe, feasible path space.

Let $\mathcal{F}$ be the continuum feasible path space and $\mathcal{F}_{free} \subset \mathcal{F}$ be the set of safe, feasible paths. Using the Green-Kelly algorithm, we sample offline from $\mathcal{F}$ a path sequence $\mathcal{P}$ of size N. At runtime, using Alg. 1, we test members of $\mathcal{P}$ in order to discover a set $\mathcal{P}_{free} \subset \mathcal{P}$ of safe paths.

The following lemma is based on the work of LaValle, Branicky, and Lindemann [16], who prove resolution completeness of deterministic roadmap (DRM) planners, which are PRM planners that draw samples from a low-dispersion, deterministic source. Since we use a deterministic sequence provided by Green-Kelly, the combination of Alg. 1 and 4 generates a DRM in path space.

Lemma 3. *For any given configuration of obstacles and any path set $\mathcal{P}_N$ generated by the Green-Kelly algorithm, there exists a sufficiently large N such that any two paths $p_i, p_j \in \mathcal{P}_{free}$ are in the same connected component of $\mathcal{F}_{free}$ if and only if Alg. 4 reports that $p_i \sim p_j$.*

Proof. LaValle, et al. [16], show that by increasing N, a sufficiently low dispersion can be achieved to make a DRM complete in any given C-Space. By an identical argument, given a continuum connected component $\mathcal{C} \subset \mathcal{F}_{free}$, all sampled paths in $\mathcal{C} \cap \mathcal{P}_N$ are in a single partition of D. If q is the radius of the narrowest corridor in $\mathcal{C}$, then for dispersion $\delta_N < q$, our discrete approximation exactly replicates the connectivity of the continuum freespace. □

Lemma 4. *Under the same conditions as in Lemma 3, there exists a sufficiently large N such that for any continuum connected component $\mathcal{C} \subset \mathcal{F}_{free}$, Alg. 1 returns a $\mathcal{P}_{free}$ such that $\mathcal{P}_{free} \cap \mathcal{C} \neq \emptyset$. That is, every component in $\mathcal{F}_{free}$ has a corresponding partition returned by Alg. 4.*

Proof. Let B_r be the largest open ball of radius r in $\mathcal{C}$. When $\delta_N < r$, B_r must contain some sample $p \in \mathcal{P}$. Since $\mathcal{C}$ is entirely collision-free, $p \in \mathcal{P}_{free}$. Thus, for dispersion less than r, $\mathcal{P}_{free}$ contains a path in $\mathcal{C}$. □

There exists a sufficiently large N such that after N samples, $\mathcal{P}$ has achieved dispersion $\delta_N < \min(q,r)$, where q and r are the dispersion required by Lemmas 3 and 4, respectively. Under such conditions, a bijection exists between the connected components of $\mathcal{P}_{free}$ and $\mathcal{F}_{free}$.

Theorem 2. *Let $D = \{\mathcal{D}_1|\ldots|\mathcal{D}_m\}$ be a partition of $\mathcal{P}_{free}$ as defined by Alg. 4. Let $C = \{\mathcal{C}_1|\ldots|\mathcal{C}_m\}$ be a finite partition of the continuum safe, feasible path space into connected components. A bijection $f : D \rightarrow C$ exists such that $\mathcal{D}_i \subset f(\mathcal{D}_i)$.*

Proof. Lemma 3 establishes that f is one-to-one, while Lemma 4 establishes that f is onto. Therefore, f is bijective. This shows that by sampling at sufficiently high density, we can achieve an arbitrarily good approximation of the connectedness of the continuum set of collision-free paths in any environment. □

Theorem 3. *A path interval p may be implicitly tested safe if it is between paths p_i and p_j such that $\mu_H(p_i, p_j) \leq d$ and a small region at the end of p_c has been explicitly tested.*

Proof. By Lemma 2, the initial interval of p_c is safe because its swath is covered by the swaths of the guard paths. Since the small interval at the end of p_c has been explicitly tested, the whole of p_c is collision-free. □

4 Results

We briefly summarize some experimental results involving equivalence class detection and implicit path collision-testing. All tests were performed in simulation on planning problems of the type described in [13].

Path classification imposes a computational overhead due to the cost of searching collision-free paths. Collision rate in turn relates to the density of obstacles in the environment. The computational overhead of our classification implementation is nearly 20% in an empty environment but drops to 0.3% in dense clutter. However, implicit collision-testing more than compensates for this overhead.

Fig. 9 shows the effect of implicit path testing on total paths tested in the absence of obstacles. As the time limit increases, the number of paths collision-tested under the traditional algorithm increases linearly at a rate of 8,300 paths per second. With implicit testing, the initial test rate over small time limits (thus small path set sizes) is over 22,500 paths per second. The marginal rate declines over time due to the aforementioned overhead, but implicit path testing still maintains its speed advantage until the entire 2,401-member path set is collision-tested.

Fig. 10 presents implicit collision-testing performance in the presence of clutter. We compare the implicit collision-tester in Alg. 5 to traditional explicit collision-testing. When fixing the replan rate at 10 Hz, implicit path evaluation maintains an advantage, despite the overhead, across all navigable obstacle densities.

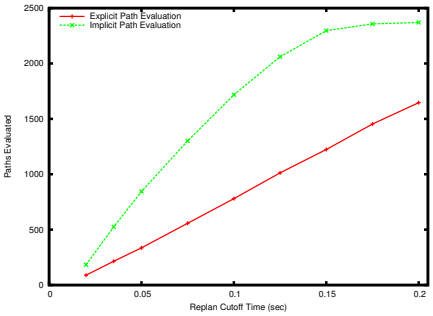


Fig. 9 Paths tested per time-limited replan step in an obstacle-free environment. Path testing performance improves by up to 3x with the algorithms we present here. Note that an artificial ceiling curtails performance at the high end due to a maximum path set of size 2,401.

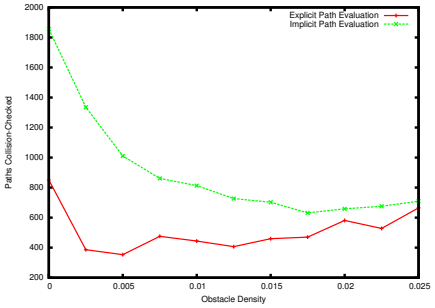


Fig. 10 Paths tested per 0.1 second time step at varying obstacle densities. Implicit collision-testing allows significantly more paths to be tested per unit time. Even in extremely dense clutter, implicit path testing considers an extra six paths on average.

5 Discussion and Future Work

In this paper, we propose an equivalence relation on local paths based on the following constraints: fixed start position and heading, fixed length, and bounded curvature. We describe an algorithm for easily classifying paths using the Hausdorff distance between them. Path classification is a tool that permits collective reasoning about paths, leading to more efficient collision-testing.

There are many other applications for path equivalence. One example uses path class knowledge in obstacle avoidance to improve visibility and safety around obstacles. Another avenue of future work involves generalizing path equivalence to higher dimensions. For instance, an implicit path test for a robot floating in 3D requires three neighboring paths, while a manipulator arm needs only two.

Acknowledgements This work is sponsored by the Defense Advanced Research Projects Agency. This work does not necessarily reflect the position or the policy of the Government. No official endorsement should be inferred. Thank you to Matthew Tesch, Laura Lindzey, and Alberto Rodriguez for valuable comments and discussions.

References

1. T. Allen, J. Underwood, and S. Scheding. A path planning system for autonomous ground vehicles operating in unstructured dynamic environments. In *Proc. Australasian Conference on Robotics and Automation*, 2007.
2. J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles. *Algorithmica*, 10(2-3-4):121–155, 1993.
3. S. Bhattacharya, V. Kumar, and M. Likhachev. Search-based path planning with homotopy class constraints. In *Proc. National Conference on Artificial Intelligence*, 2010.

4. H. Blum. A transformation for extracting new descriptors of shape. In Weiant Whaters-Dunn, editor, *Proc. Symposium on Models for the Perception of Speech and Visual Form*, pages 362–380, Cambridge, Mass., 1967. MIT Press.
5. H. Choset and J. Burdick. Sensor based planning, part I: The generalized Voronoi graph. In *Proc. International Conference on Robotics and Automation*, pages 1649–1655, 1995.
6. Michael Farber. Topological complexity of motion planning. *Discrete & Computational Geometry*, 29(2):211–221, 2003.
7. N. H. Gardiol and L. P. Kaelbling. Action-space partitioning for planning. In *National Conference on Artificial Intelligence*, Vancouver, Canada, 2007.
8. C. Green and A. Kelly. Toward optimal sampling in the space of paths. In *Proc. International Symposium of Robotics Research*, Hiroshima, Japan, November 2007.
9. J. Henrikson. Completeness and total boundedness of the Hausdorff metric. *The MIT Undergraduate Journal of Mathematics*, 1, 1999.
10. L. Jaillet and T. Simeon. Path deformation roadmaps: Compact graphs with useful cycles for motion planning. *International Journal of Robotics Research*, 27(11–12):1175–1188, 2008.
11. L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. In *Proc. International Conference on Robotics and Automation*, pages 566–580, 1996.
12. O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proc. International Conference on Robotics and Automation*, St. Louis, USA, March 1985.
13. R. A. Knepper and M. T. Mason. Empirical sampling of path sets for local area motion planning. In *Proc. International Symposium of Experimental Robotics*, Athens, Greece, July 2008.
14. R. A. Knepper, S. S. Srinivasa, and M. T. Mason. Curvature bounds on the weighted Voronoi diagram of two proximal paths with shape constraints. Technical Report CMU-RI-TR-10-25, Robotics Institute, Carnegie Mellon University, 2010.
15. J. P. Laumond. Feasible trajectories for mobile robots with kinematic and environment constraints. In *Intelligent Autonomous Systems, An International Conference*, Amsterdam, The Netherlands, December 1986.
16. S. M. LaValle, M. S. Branicky, and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotics Research*, 23(7/8):673–692, July/August 2004.
17. S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, May 2001.
18. V. Lumelsky and A. Stepanov. Automaton moving admist unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.
19. E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige. The office marathon: Robust navigation in an indoor office environment. In *Proc. International Conference on Robotics and Automation*, May 2010.
20. J. R. Munkres. *Topology*. Prentice Hall, Upper Saddle River, NJ, 2000.
21. H. Niederreiter. *Random Number Generation and Quasi-Monte-Carlo Methods*. Society for Industrial Mathematics, Philadelphia, 1992.
22. J. Reif and H. Wang. The complexity of the two dimensional curvature-constrained shortest-path problem. In *Third International Workshop on Algorithmic Foundations of Robotics*, pages 49–57, June 1998.
23. P. Sampl. Medial axis construction in three dimensions and its application to mesh generation. *Engineering with Computers*, 17(3):234–248, 2001.
24. G. Sánchez and J.-C. Latombe. On delaying collision checking in PRM planning: Application to multi-robot coordination. *International Journal of Robotics Research*, 21(1):5–26, 2002.
25. E. Schmitzberger, J.L. Bouchet, M. Dufaut, D. Wolf, and R. Husson. Capture of homotopy classes with probabilistic road map. In *Proc. International Conference on Intelligent Robots and Systems*, October 2002.
26. M. Vendittelli, J. P. Laumond, and C. Nissoux. Obstacle distance for car-like robots. *IEEE Transactions on Robotics and Automation*, 15:678–691, 1999.
27. C. K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete & Computational Geometry*, 2:365–393, 1987.