COMPUTING PARETO OPTIMAL COORDINATIONS ON ROADMAPS

ROBERT GHRIST, JASON M. O'KANE, AND STEVEN M. LAVALLE

ABSTRACT. We consider coordination of multiple robots in a common environment, each robot having its own (distinct) roadmap. Our primary contribution is a classification of and exact algorithm for computing vector-valued — or **Pareto** — optima for collision-free coordination. We indicate the utility of new geometric techniques from CAT(0) geometry and give an argument that curvature bounds are the key distinguishing feature between systems for which the classification is finite and for those in which it is not.

1. INTRODUCTION

Managing a collection of automated guided vehicles (or AGVs) in a common workspace is a canonical example of a coordination problem in robotics. Such coordination problems introduce challenges beyond those of simple obstacle avoidance and motion planning in single robot settings; robot-robot interactions must likewise be controlled.

In keeping with contemporary approaches to motion planning, we assume that each robot possesses a precomputed roadmap approximation to its individual configuration space. We do not, however, assume that there is a single roadmap for the entire multi-agent configuration space: each robot has a roadmap which takes account of workspace obstacles but not of the other robots.

Given this structure as an input, we consider the problem of optimal coordination with respect to elapsed time (for concreteness; other cost functions are amenable to analysis). The appropriate notion of optimality can differ from one application to another. Settings in which average elapsed time or total elapsed time are common and often appropriate (e.g., [21, 29, 38]). However, all such approaches scalarize the N distinct pieces of cost function data.

Our perspective is to emphasize vector-valued optimization, preserving all cost function data. This notion of **Pareto optimality** [32, 35] is widely used in mathematical economics to model individual consumers striving to optimize distinct economic goals. It is more "faithful" to the system in the sense that no data is lost by scalarization.

Research supported in part by NSF PECASE Grant # DMS - 0337713 and NSF IIS-0296126 [JO,SL].

It is to be noted in particular that a successful classification of Pareto optima has several benefits. First, it automatically yields the set of all optima for all (monotone) scalarizations of the cost functions: see Lemma 1.1. Second, it allows the user to have a (hopefully small, finite) template of optimal coordinations which can be used for on-line adaptation to changing needs and cost functions.

1.1. **History.** The problem of coordinating robots along fixed roadmaps can be considered as a special case of general motion planning for multiple robots. Previous approaches to multiple-robot motion planning are often categorized as *centralized* or *decoupled*. A centralized approach typically constructs a path in a composite configuration space, which is derived from the Cartesian product of the configuration spaces of the individual robots (e.g., [3, 4, 36]). A decoupled approach typically generates paths for each robot independently, and then considers the interactions between the robots (e.g., [8, 15, 31]).

The approach in [15] prioritizes the robots, and defines a sequence of planning problems for which each problem involves moving one robot while those with higher priority are considered as predictable, moving obstacles. This involves the construction of two-dimensional path-time space [24] over which the velocity of the robot is tuned to avoid collisions with the moving obstacles.

In [2, 6, 10, 34, 31, 37] robot paths are independently determined, and a coordination diagram is used to plan a collision-free trajectory along the paths. The approaches in [2, 34] additionally consider dynamics. In [26, 40], an independent roadmap is computed for each robot, and coordination occurs on the Cartesian product of the roadmap path domains. The suitability of one approach over the other is usually determined by the trade-off between computational complexity associated with a given problem, and the amount of completeness that is lost. In some applications, such as the coordination of AGVs, the roadmap might represent all allowable mobility for each robot.

In this paper, we focus on the multiplicity of Pareto optimal path coordinations among cylindrical obstacles — those determined by pairwise collisions (see §1.4 below). In [27], an approximate Dijkstra-like algorithm to find Pareto optimal solutions in this context was given. To our knowledge, the only previously known exact solution is that of [12], which applies only to the case of two robots that translate on acyclic roadmaps.

1.2. **Coordination spaces.** We assume that there are N robots, each having a roadmap Γ_i (a graph within the C-space of the i^{th} robot) precomputed independent of the other robots. A **roadmap coordination space** of $\{\Gamma_i\}_1^N$ is a deleted product of the roadmaps:

$$\mathcal{X} := (\Gamma_1 \times \cdots \times \Gamma_N) - \mathcal{O}, \tag{1.1}$$

where $\mathcal{O}$ denotes an (open) obstacle set.

Throughout the paper, all coordination spaces are assumed to be *piecewise linear (PL) manifolds*. This assumption rules out obstacle sets $\mathcal{O}$ which are too intricate to have a nice local structure or for which the system "locks up" in a singular configuration. Certain results of this paper hold under much weaker technical assumptions (including piecewise analytic or algebraic) [16].

A special example of a roadmap coordination space arises when all of the roadmaps Γ_i are identical, and the obstacle set is an open neighborhood of $\{x_i = x_j : \text{for some } i \neq j\}$. In this case, one can consider the workspace to be the graph itself, and the roadmap coordination space is precisely the configuration space of N labeled objects on the graph.

This example is particular in that all obstacles are defined by pairwise constraints on positions. More generally, we say that the obstacle set $\mathcal{O}$ for $\mathcal{X} = \prod \Gamma_i - \mathcal{O}$ is **cylindrical** if, for some collection of sets $\{\Delta_{i,j} \subset \Gamma_i \times \Gamma_j\}_{i<j}$, it holds that $\mathcal{O}$ is of the form

$$\mathcal{O} = \left\{ (x_k)_1^N \in \prod_k \Gamma_k \ : \ (x_i, x_j) \in \Delta_{i,j} \ \text{for some } i < j \right\}. \tag{1.2}$$

Physically, this means that whenever two robots experience an obstacle (*e.g.*, collision), the states of all other robots are irrelevant. For the algorithms in this paper, we require each $\Delta_{i,j}$ to be the interiors of disjoint PL subsets of $\Gamma_i \times \Gamma_j$; the mathematical results we state hold in a much broader setting [16].

One of the important perspectives of this paper is that there is a significant difference between computing optima for cylindrical versus non-cylindrical coordination spaces.

1.3. **Pareto optimality.** Pareto optimization is simply vector-valued optimization. In the context of robotics applications, Pareto optimization arises when distinct robots possess distinct goals and/or cost functions for evaluating performance. Each robot wishes to optimize its cost function independently of the others.

Mathematically, this is characterized as follows. Given a parameterized path $\gamma : [0, T] \to \mathcal{X}$ in a coordination space, each robot executes the projected path $\gamma_i := P_i \circ \gamma$, where P_i denotes projection onto the i^{th} factor. Given cost functions $\{\tau_i\}_1^N$, the cost vector for γ is the vector $\tau(\gamma) := (\tau_i(\gamma_i))_1^N$.

For the remainder of this paper, we assume that each roadmap Γ_i is outfitted with a metric such that the speed of the i^{th} robot traveling along this graph at maximal speed is exactly one; hence all admissible paths have speeds whose components are bounded above by one. It will be assumed for simplicity that the cost functions τ_i agree with elapsed time. This is an important and characteristic example. More general cost functions are often interesting and important; we believe that a simple extension of our methods covers such cases.

A path $\gamma : [0, T] \to \mathcal{X}$ is **Pareto optimal** iff $\tau(\gamma)$ is minimal with respect to the partial order on vectors:

$$\tau(\gamma) \leq \tau(\gamma') \Leftrightarrow \tau_i(\gamma_i) \leq \tau_i(\gamma_i') \, \forall \, i. \tag{1.3}$$

The Pareto optima comprise the set of all optima for all monotone scalarizations (such as, *e.g.*, average time-to-goal and all non-linear generalizations thereof):

Lemma 1.1. *For any scalarization $f : \mathbb{R}^n \to \mathbb{R}$ with $\partial f / \partial x_i > 0$, all minima of $f \circ \tau$ are Pareto optima.*

Proof: Given any minimal path for $f \circ \tau$ which is not Pareto optimal, deforming it to a Pareto optimal path decreases some τ_i without increasing any of the others; hence it decreases the f-value: contradiction. □

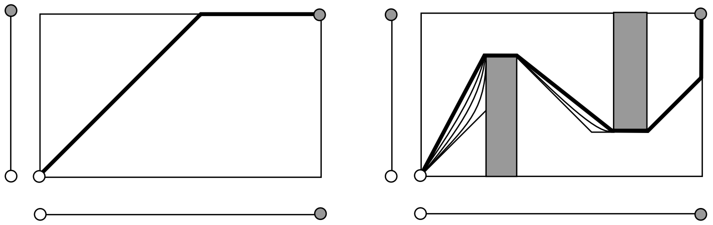

FIGURE 1. [left] The unique Pareto optimal path on a rectangle from the lower-left to the upper-right corners; [right] an envelope of Pareto optimal paths weaving through obstacles forms a single equivalence class.

Pareto optimal paths are rarely unique. Two paths γ and γ' are **Pareto equivalent** iff they are homotopic through Pareto optimal paths which are equal in the partial order; *i.e.*, $\tau(\gamma) = \tau(\gamma')$. Fig. 1[right] illustrates a single Pareto optimal class with many representatives.

1.4. **Contributions.** This paper does three things:

(1) We state a uniqueness result for locally Pareto optimal classes on a cylindrical coordination space which depends only on the topology of the space, and not on its geometry or PL structure.
(2) We give an exact algorithm for computing Pareto optima on such cylindrical coordination spaces.
(3) We propose an explanation for why cylindrical coordination spaces are "easy" while non-cylindrical spaces can be "hard." This explanation is based on curvature bounds.

The first contribution — the uniqueness result — is given in Section 2. In brief, every path in a cylindrical coordination space between fixed endpoints is homotopic (fixing the endpoints) to a unique class which is Pareto optimal among paths in its homotopy class. From this result, one can show that only a finite number of these local optima are in fact global. The proof of this result is outlined in Section 3.2. A complete proof (with extensions) is given in a more technical paper [16].

The second contribution — the exact algorithm — is the most technical portion of the paper. The algorithm takes as its input the coordination space, the endpoints of the path, and the homotopy class of the locally Pareto optimal path desired. In Section 4 we give an exact algorithm under the reasonable assumption that robots do not backtrack. The algorithm is output sensitive with respect to the complexity of the path it computes. Fix the number of robots; let m denote the complexity of the obstacle set $\mathcal{O}$; and let p and p' denote the complexities of the input and output paths respectively. Algorithm 1 executes in $O(p + m \log mp + p'\sqrt{m} \log m)$ time. An extension to arbitrary paths involving backtracking is also given, along with examples in both cases.

The final contribution — an explanation for why these results hold — focuses on notions of synthetic curvature as arises in Alexandrov geometry and geometric group theory. This perspective is novel in robotics, but has a rich mathematical heritage and an impressive list of achievements. Background material is outlined in Section 3.1 with applications to the present problem sprinkled throughout the text.

2. CLASSIFYING LOCAL OPTIMA

2.1. **Examples of Pareto optimal classes.** We illustrate two key examples of simple coordination spaces and Pareto optimal path classes.

Example 2.1. Consider the case where $N = 2$, $\Gamma_1 = \Gamma_2 = [-2, 2]$, and $\mathcal{O} = \{(x, y) : x^2 + y^2 < 1\}$, which corresponds to a pair of identical disc-shaped AGV's sliding along interval roadmaps which intersect in the workspace at right angles. There are exactly two Pareto optimal classes of paths from $(-2, -2)$ to $(2, 2)$, as illustrated in Fig. 2[left]. The difference between these two paths lies in which robot decides to pause in order to allow the other to pass through the intersection.

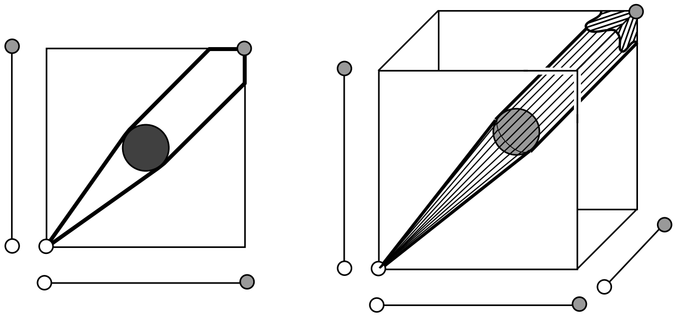

FIGURE 2. [left] The case of two robots with a disc obstacle possesses two Pareto optimal classes; [right] Three robots with a spherical obstacle possesses a continuum of Pareto optimal classes.

Example 2.2. We modify the previous example by letting $N = 3$ and choosing $\mathcal{O}$ to be a round open ball of radius 1 at the origin. By the symmetry of $\mathcal{X}$ about the diagonal of the cube, it is clear that there is a circle's worth of paths which begin at $(-2,-2)$, trace a straight line which is tangent to $\mathcal{O}$, and then exit this sphere tangentially with slope one. The projection of this family of paths to the first two coordinates includes as special cases the distinct Pareto optima of Example 2.1, as well as a continuum of paths whose goal times interpolate between these two. Hence, there is a continuum of Pareto optimal classes.

Neither of these examples is PL. In the case of Example 2.1, any PL approximation to the coordination space yields exactly two Pareto optimal classes. In the case of Example 2.2, PL approximations to the space may possess only a finite number of Pareto optimal classes. However, the number of such classes grows with the resolution of the PL approximation, approaching infinity in the limit as the obstacle approaches a smooth ball.

These two examples are key to understanding the results of this paper. Notice that the former is cylindrical (all 2-d coordination spaces necessarily are) while the latter is not. Notice also that in the cylindrical case, the two Pareto optima are in different homotopy classes of path (one is not deformable to the other fixing the endpoints). On the other hand, in the non-cylindrical example, all of the numerous, distinct Pareto optima are homotopic as paths with fixed endpoints.

2.2. **A classification statement.** In [16] the following theorem is proved.

Theorem 2.3. *Let $\mathcal{X}$ denote a cylindrical coordination space and let $\mathcal{P}$ denote a homotopy class of paths on $\mathcal{X}$ with fixed endpoints. Then among all paths of $\mathcal{P}$ there exists a unique Pareto optimal equivalence class.*

We call such a class a **locally** Pareto optimal class, since they are Pareto optimal only with respect to a fixed homotopy class. If one "cuts" the path and reconnects in a different manner, the local Pareto optimum in this class may be a global improvement. It follows that locally Pareto optimal paths on a cylindrical coordination space are in bijective correspondence with homotopy classes of paths.

As such, the number of global Pareto optima is highly restricted, in sharp contrast to Example 2.2. For example, if one should have a simply connected coordination space $\mathcal{X}$, then there exists a unique Pareto optimal class: remarkably, this one class of paths minimizes any monotone scalarization of the goal times. More generally, Theorem 2.3 is the key to proving the following finiteness result [16]:

Theorem 2.4. *On any cylindrical coordination space, there exists a finite number of Pareto optimal classes.*

The intuition for this result is that, via Theorem 2.3, there is one locally Pareto optimal class per homotopy class of paths. In the case where there is an infinite number of homotopy classes of paths, all but a finite number of these "wrap" around a hole in the space multiple times. This wrapping is wasteful — it forces each robot to expend more time to reach its goal — and hence is strictly worse than a homotopy class which does not wind about. (cf. Example 2.1, in which additional loops about the central hole would certainly be wasteful.)

With this finiteness result, it is reasonable to tabulate representatives of the Pareto optimal classes for a given system. The problem remains, however, which representatives to use and how to compute them. This is the subject of the remainder of the paper. The techniques used to prove Theorem 2.3 are necessary for answering both questions. As such, we present the requisite mathematical background for utilizing these tools, followed by the algorithms used to compute representatives.

3. CURVATURE AND OPTIMAL COORDINATION

3.1. CAT(0) **geometry.** There are many notions of curvature applicable to a space outfitted with a geometry (a Riemannian metric): scalar curvature, section curvature, Ricci curvature, etc. Discrete curvatures for PL domains have also been explored. But the notion of curvature that is used in the proof of Theorem 2.3 is quite a bit more general.

Let X denote a metric space (a space outfitted with a means of measuring distances between points) for which the notion of a geodesic makes sense: that is, for any pair of points $p, q \in X$, there exists a path γ from p to q whose length is equal to the distance between p and q. It is not assumed that this path is unique. All the coordination spaces we consider are Euclidean domains and hence satisfy this property [7].

For such spaces as this, an old notion of curvature (developed by, among others, Alexandrov and Toponogov) uses triangles to bound curvature from above. Consider three points p, q, and r in X and three shortest paths $\overline{pq}$, $\overline{qr}$, and $\overline{pr}$ in X forming a geodesic triangle T. In the Euclidean plane, the sum of the three angles of T would equal exactly π. However, if this triangle were drawn on the surface of a round sphere, the sum of the angles would be strictly larger than π. Conversely, a geodesic triangle drawn on a hyperbolic plane (a "saddle" surface) would have the sum of the angles strictly less than π.

Roughly speaking, one says that X is CAT(0) if for every geodesic triangle between any set of points p, q, r, the sum of the angles is no greater than π. For example, all simply connected subsets of the Euclidean plane are CAT(0) . Furthermore, all convex sets of arbitrary dimension are likewise CAT(0) ; thus, the CAT(0) condition is a generalization of both two-dimensionality and convexity.

To make this precise for a general space X requires a little more care: angles are not so easily measured, since our only assumption on X is that shortest paths exist. One proceeds by **comparison triangles**.[1] For each triple of points $p, q, r \in X$, draw the triangle in X with geodesics $\overline{pq}$, $\overline{qr}$, and $\overline{rp}$. Let p', q', and r' denote three points in the Euclidean plane forming a triangle whose respective edges equal in length those in X. Choose a pair of points s and t on the edges of the geodesic triangle in X and consider the corresponding points s' and t' in the plane. The CAT(0) **inequality** for X is the following: for every p, q, r in X, and for every s, t, one has $d(s,t) \leq \|s' - t'\|$, where $\|\cdot\|$ denotes the Euclidean norm in the plane: see Figure 3. A geodesic space X is CAT(0) if the CAT(0) inequality holds for each geodesic triangle in X.

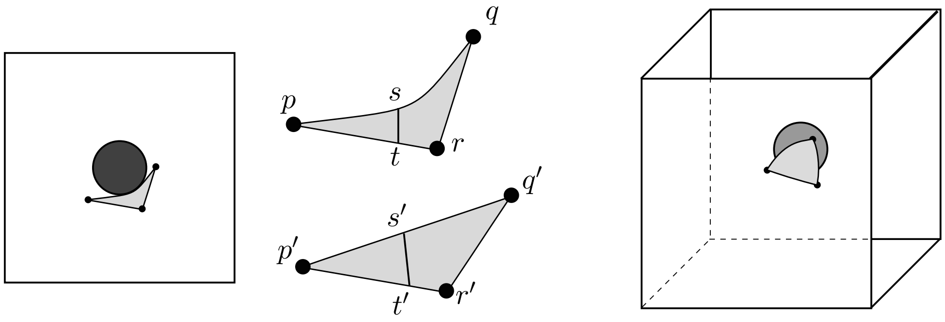

FIGURE 3. The coordination space of Example 2.1 is locally CAT(0) [left] since small triangles are no "fatter" than their planar counterparts [center], whereas that of Example 2.2 is not locally CAT(0) [right].

One says that the space X is **nonpositively curved** if it is locally CAT(0) — that is, if sufficiently small geodesic triangles have total angle no greater than π. All subsets of the Euclidean plane are nonpositively curved, regardless of whether they are simply connected or not. In fact, it is easy to show that a space X is CAT(0) if and only if it is both nonpositively curved and simply connected.

The application of CAT(0) geometry to coordination spaces begins with the following important result [16]:

Lemma 3.1. *Cylindrical coordination spaces are nonpositively curved.*

The proof of this result requires techniques from CAT(0) geometry which fall outside the scope of this paper: see [16] for a complete proof.

3.2. **Geodesics and left-greedy paths.** A simple result about CAT(0) spaces with important applications is the following:

Lemma 3.2. *Between any two points in a* CAT(0) *space X there exists a unique geodesic.*

[1] The term CAT(0) comes from the mathematicians' names Cartan, Alexandrov, Toponogov. However, it can also be thought of as *comparare ab triangulos* — to compare from triangles.

The result is standard. We include a proof for the sake of exposition.

Proof: Let $p, q \in X$ be points for which there are two shortest paths γ and γ'. Choose a point r at the midpoint of the path γ'. Then the two halves of γ' are geodesic paths from p to r and r to q respectively. Thus, γ and γ' together give a geodesic triangle in X. Drawing the comparison triangle in the Euclidean plane yields a degenerate line segment, since γ and γ' have the same length. The CAT(0) inequality implies that $\gamma = \gamma'$. □

Corollary 3.3. *In any fixed homotopy class of paths between points in a space of nonpositive curvature, there exists a unique shortest path.*

Proof: This uses a simple construction from topology which will be useful in the remainder of the paper. Given a connected space X, the **universal cover** of X is a connected and simply-connected space $\tilde{X}$ together with a projection map $P : \tilde{X} \rightarrow X$ which is a homeomorphism on sufficiently small open sets. For example, the universal cover of the circle is the real line, with the projection map being $P : x \rightarrow e^{2\pi i x}$. Any space X with which we are concerned possesses a universal cover which is unique up to a type of symmetry: see [19, Ch. 1] for a complete treatment.

The universal cover of any nonpositively curved space is a CAT(0) space under the lifted metric [7]. Given points $p, q \in X$ and a corresponding point $\tilde{p} \in P^{-1}(p)$, any fixed homotopy class of paths in X from p to q corresponds to the set of all paths between $\tilde{p}$ and some fixed $\tilde{q} \in P^{-1}(q)$. The projection P preserves lengths of paths since P locally preserves lengths. Lemma 3.2 thus implies that there is a unique shortest path on $\tilde{X}$. □

This does not imply Theorem 2.3, but is rather a prototype. The heart of Theorem 2.3 is in showing that a certain type of path in $\mathcal{X}$ (here called a **left-greedy** path) is unique up to homotopy. Recall Figure 1[left] — the unique geodesic in this case (the diagonal) is not Pareto optimal. Both the shortest path and the Pareto optimal path are geodesics in the ℓ^∞ metric: these minimize total elapsed time of the system. The set of ℓ^∞ geodesics on X between fixed endpoints is almost always a continuous envelope of paths. The Pareto optimal path in Figure 1[left] is the path on the "left" edge of the envelope. This is the canonical example of a left-greedy path.

The definition of left-greedy paths in [16] uses a discrete version, based on a cubical approximation and a result by Niblo and Reeves on **normal cube paths** in CAT(0) cubical complexes [30]. In order to avoid unpleasant details about spatial discretization, we give an equivalent "smooth" version which is better suited to the algorithms of the next section.

Assume first that $\mathcal{X}$ is a simply connected cylindrical coordination space with $p, q \in \mathcal{X}$. Let γ be a path from p to q. For any point $y = (y_k)_1^N \in \mathcal{X}$ on the path γ, consider the N distinct **hyperplanes** at y: $\mathcal{H}_k(y)$ is defined to be the connected component of $\{x \in \mathcal{X} : x_k = y_k\}$. Since $\mathcal{X}$ is CAT(0) , hyperplanes always separate $\mathcal{X}$.

We say that a path from q to p is **left-greedy** if it crosses hyperplanes between p and q as quickly as possible. More specifically, for any $y \in \gamma$ and all $k = 1 \dots N$, the tangent vector to the path, $\dot{\gamma}(y)$, satisfies the following:

(1) If $\mathcal{H}_k(y)$ separates p from q in $\mathcal{X}$, then the k^{th} component of $\dot{\gamma}(y)$ is nonzero and is positive/negative so as to point from p to q;
(2) If $q \in \mathcal{H}_k(y)$, then the k^{th} component of $\dot{\gamma}(y)$ is zero;
(3) All components of $\dot{\gamma}(y)$ are maximized with respect to the speed constraints of 1 and the obstacle constraints.

In the case where $\mathcal{X}$ is not simply connected, one extends the definition of left-greedy by lifting $\mathcal{X}$ to the universal cover (see proof of Corollary 3.3) and working with paths on this simply connected space as above. The following result is proved in [16], using Lemma 3.1 and a result of Niblo and Reeves [30]:

Theorem 3.4. *There exists a unique left-greedy path in each homotopy class of a cylindrical coordination space.*

The proof of Theorem 2.3 consists of showing that any Pareto optimal path is Pareto equivalent to the left-greedy path in its homotopy class.

4. ALGORITHMS

We now consider the algorithmic problem of computing exact Pareto optimal paths. In this section, we give an efficient algorithm for deforming a path to its Pareto optimal representative. The input to the algorithm is a collision-free path γ; the output is the Pareto optimal path γ' homotopic to γ. We know from Section 3.2 that it suffices to compute the unique left-greedy path homotopic to γ.

The central idea is to take advantage of the cylindrical structure of the obstacle set. Indeed, $\mathcal{X}$, which is itself N-dimensional, can be fully described by a set of $\binom{N}{2}$ projections, each containing only the pairwise collision regions for two of the robots. More precisely, we define the projection $\mathcal{X}_{i,j}$ for $\mathcal{R}_i$ and $\mathcal{R}_j$ as

$$\mathcal{X}_{i,j} = (\Gamma_i \times \Gamma_j) - \Delta_{i,j}. \tag{4.1}$$

From such projections the original space may be realized by extruding the obstacles across the remaining $N - 2$ dimensions. We exploit this structure by constructing γ' directly in $\mathcal{X}$ using geometric operations only on the projections.

To simplify the presentation, we first describe an algorithm for the case in which the robots cannot backtrack along their paths, then outline the extension to allow backtracking. The former appears in Section 4.1; the latter in Section 4.2. Finally, Section 4.3 discusses the complexity of computing the set of all Pareto optimal solutions.

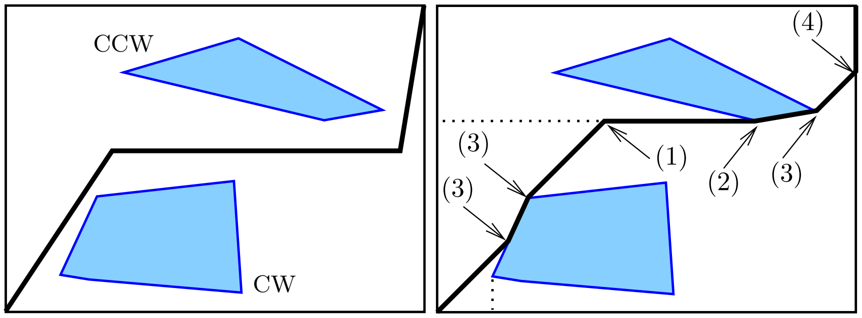

FIGURE 4. [left] A sample 2-d coordination space and a collision-free path within it. [right] The left-greedy deformation of this path, showing each of the 4 types of velocity changes described in the text. Dashed lines indicate the critical events. The deformed path γ' cannot cross any critical event.

4.1. **Computing monotone Pareto optimal paths.** A path is **monotone** if it is nondecreasing in each dimension. Monotonicity is equivalent to the requirement that robots may not "back up" along their paths. Throughout Section 4.1, we assume that the given γ can be deformed into a monotone path.

In a coordination space with PL obstacles, left-greedy coordinations will themselves be PL. Our algorithm builds a path by starting with a single point at the initial position and adding linear path segments sequentially until the goal is reached. Each of these linear path segments corresponds to a portion of the coordination in which all of the robots maintain constant velocity. Changes of velocity can be characterized in one of four ways (as illustrated in Figure 4):

(1) stopping to avoid a future collision (when a projection of the path reaches a horizontal or vertical minimum of some obstacle around which it may not go without changing the homotopy class of the original path),
(2) restarting from case 1 when the obstacle has been reached,
(3) stopping, slowing or speeding up when the path initiates or breaks contact with some coordination space obstacle, and
(4) stopping each robot that reaches its goal.

Each iteration of the algorithm has three parts: determining which of these events will occur next, adding a segment to the output coordination advancing to this point, and determining the left-greedy velocity at which to continue. The method is summarized in Algorithm 1; elaboration follows.

4.1.1. *Next Event Computation.* In each projection, a monotone path will appear as a polygonal chain without self-intersection connecting the lower left to the upper right of the projection. Such a projected path will divide the projection into two parts – one

Algorithm 1 ExactLeftGreedyMonotone($\mathcal{X}, \gamma$)

```
Require: $\gamma$ is a collision-free coordination in $\mathcal{X}$ whose left-greedy representative is monotone.
$x_0 \leftarrow x_{init}$
$W \leftarrow \emptyset$
$\mathcal{O} \leftarrow$ ObstacleRegions($\mathcal{X}$)
$R \leftarrow$ ObstacleOrientations($\mathcal{O}, \gamma$)
$E \leftarrow$ CriticalEvents($\mathcal{O}, \gamma$)
$t \leftarrow 0$
while $x_t \neq x_{goal}$ do
    $t \leftarrow t + 1$
    $v \leftarrow$ MaximalVelocity($x, W, \mathcal{O}$)
    $e \leftarrow$ NextEvent($x, v, R, E, \mathcal{O}$)
    $W \leftarrow$ UpdateWaitForSet($p, e, W$)
    $x_t \leftarrow x_{t-1} + v \cdot e$.time
end while
return $(x_0, \ldots, x_t)$
```

above γ and one below. We label each obstacle according to this subdivision. Obstacles above γ receive a counterclockwise (CCW) label; obstacles below γ are labeled clockwise (CW). A crucial observation is that if γ is homotopic to some monotone path, then these orientations fully identify the homotopy class of γ. Therefore if we construct γ' in such a way that it induces the same orientations on the obstacles as γ, we can be certain that γ' is homotopic γ.

To assign an orientation to an obstacle in $\Delta_{i,j}$, we need to determine whether it is above or below γ in $\mathcal{X}_{i,j}$. Since γ divides $\mathcal{X}_{i,j}$ into two regions with PL boundaries, computing the orientation for each obstacle is a planar point location problem. Let m denote the total complexity of the obstacle regions. Then these point location queries can be solved for all obstacles in optimal $O(p + m \log p)$ time using recursive triangulation methods [25].

One concrete way to think of these orientations is as a priority scheme for scheduling the robots. Consider an obstacle in $\mathcal{X}_{i,j}$. This obstacle represents a region that cannot be occupied by both $\mathcal{R}_i$ and $\mathcal{R}_j$ at once. The orientation induced by γ determines a priority for these robots with respect to this collision region. A CW orientation dictates that $\mathcal{R}_j$ should pass through this collision region first and as a result, when $\mathcal{R}_i$ reaches the boundary of the collision region, it should stop and wait for $\mathcal{R}_j$ to pass through. The reverse is true for CCW-oriented obstacles – $\mathcal{R}_j$ should wait at the boundary of the collision region for $\mathcal{R}_i$ to pass.

To facilitate building γ' with orientations identical to γ, we extend a ray downward from the horizontal minimum of each CW obstacle. These rays mark the "last chance" for a monotone path to pass over these obstacles without breaking homotopy with γ. Similarly, leftward rays from vertical minima of CCW obstacles indicate the final opportunities to pass under these obstacles. We call the collection of all such rays a set of "critical events." When γ reaches a critical event in some $\mathcal{X}_{i,j}$, the corresponding robot should be stopped, preventing γ' from crossing the critical event. As the algorithm proceeds, it maintains a set W of ordered triples (i, j, a) in order to track which robots are stopped in this way and when each is permitted to restart. An element $(i, j, a) \in W$ indicates that $\mathcal{R}_i$ is stopped waiting for $\mathcal{R}_j$ to reach the point a along Γ_j. Elements are added to W at events of type 1 and removed at events of type 2.

Now we can describe the NEXTEVENT procedure from Algorithm 1. Computing the next event involves a ray-shooting query in each projection. In $\mathcal{X}_{i,j}$, extend a ray starting from the current position in the direction determined by ratio of velocities for $\mathcal{R}_i$ and $\mathcal{R}_j$. The first critical event or obstacle intersected by this ray represents the next event in $\mathcal{X}_{i,j}$. This computation can be done in time $O(\sqrt{m}\log m)$ using the algorithm of [11]. To find the next event for the entire N-robot system, repeat this process across all $\binom{N}{2}$ projections and select the earliest event from among these.

4.1.2. *Computing local left greedy velocities.* Constructing a left greedy path implies that each robot should maintain the fastest collision-free velocity consistent with W, which may constrain some robots to have zero velocity. Given W and a point x in coordination space, we want to compute a vector $v = (v_1, \ldots, v_N)$ of velocities. In this section we will show that this is equivalent to finding the maximum feasible velocity for each robot independently and describe a single linear program that finds all of these maxima.

The velocity vector v is constrained in four ways:

(1) Global velocity constraints induced by our model: $0 \le v_i \le 1$ for each $0 \le i \le N$.
(2) For each $(i, j, a) \in W$, $v_i = 0$.
(3) For each obstacle boundary edge e containing x, let $\mathcal{X}_{i,j}$ be the projection containing e and let m denote the slope of e. Then in order to prevent an immediate collision, we have the constraint

$$v_i - mv_j \le 0 \tag{4.2}$$

if x is below the obstacle or

$$v_i - mv_j \ge 0 \tag{4.3}$$

if x is above the obstacle.
(4) For each robot $\mathcal{R}_i$ that has reached its goal, $v_i = 0$.

It remains for us to determine what objective function to maximize. Consider the family of linear programs $L_1, \ldots, L_N$ with identical constraint sets defined by (1)-(4) above and in which the objective in L_i is to maximize v_i. Let $L_i(j)$ denote the value assigned to v_j by an optimal solution to L_i. Observe that $L_i(i)$ is the maximum feasible value for v_i.

Lemma 4.1. *Selecting* $v = (L_1(1), L_2(2), \ldots, L_N(N))$ *satisfies (1)-(4).*

Proof: Suppose some constraint is violated. Constraints of forms (1), (2) and (4) are certainly satisfied. Without losing generality, choose i and j so that $v_i - mv_j \leq 0$ is a violated constraint of form (3), so that $L_i(i) - mL_j(j) > 0$. Since $L_j(j)$ is the maximum feasible value for v_j, we have $L_i(j) \leq L_j(j)$. In a monotone path we will have $m \geq 0$, so we may substitute to find $L_i(i) - mL_i(j) > 0$, violating a constraint in L_i and contradicting the definitions of $L_i(i)$ and $L_i(j)$. □

Finally, note that $(L_1(1), L_2(2), \ldots, L_N(N))$ is the optimal solution to the linear program with objective function $\sum_i v_i$. Thus, we can compute the desired velocity by solving a single linear program in N variables and $O(N^2)$ constraints. For fixed N, such a program can be solved in constant time. In practice, the number of constraints is the sum of the cardinality of W and the number of obstacles x touches; this value can be expected to be reasonably small.

4.1.3. *Analysis.* Recall that N denotes the number of robots and m the total complexity of the pairwise obstacle regions. Let denote p the number of linear segments in γ. We consider N fixed. Our algorithm is output sensitive in the sense that the run time is a function of the complexity p' of the locally Pareto optimal path generated.

We can generate the critical events using a straightforward generalization of the standard vertical decomposition algorithm in [13] in time $O(m \log m)$. Each iteration of loop on lines 6-12 adds an additional segment to γ'. We have already argued that each such iteration can be completed in time $O(\sqrt{m} \log m)$. The preprocessing steps of generating critical events and assigning obstacle orientations takes time $O(p + m \log m + m \log p)$. Therefore, Algorithm 1 executes in $O(p + m \log mp + p' \sqrt{m} \log m)$ time in total.

Finally, we bound p'. The number of velocity changes of types 1, 2, and 3 can each be bounded by m; there are exactly N events of type 4. Therefore $p' < 3m + N$ and we may state the time complexity of the algorithm for fixed N as $O(p + m \log mp + m^{\frac{3}{2}} \log m)$.

4.1.4. *Computed Examples.* We have implemented a simplified version of Algorithm 1. In particular, we perform both the point-location step and the next event selection using the obvious quadratic time algorithms. Our implementation is in C++ on Linux and experiment times shown are for a 2.55GHz processor. Figure 5 illustrates a simple coordination problem with $n = 3$ for which we found the set of three Pareto optima by exhaustively enumerating monotone homotopy classes. These coordinations took approximately 0.3 seconds to compute. Figure 7 shows two more complex coordination problems solved by our implementation – one in which 8 robots translate left-to-right, and a "swap" problem in which 20 robots switch sides.

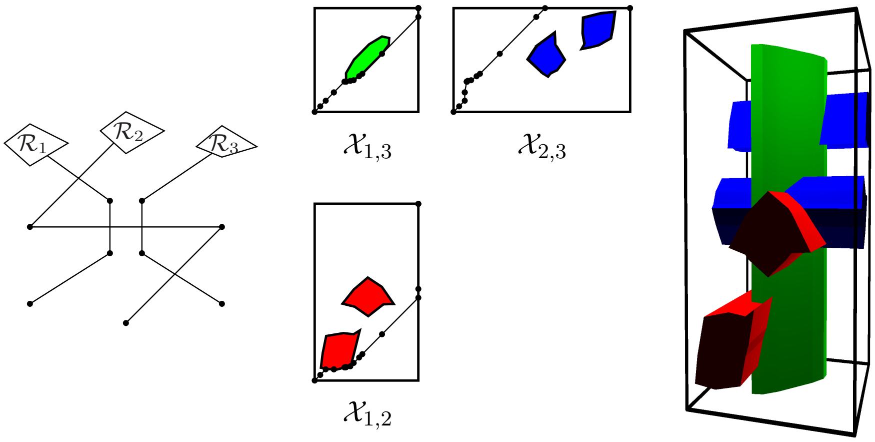

FIGURE 5. [left] A coordination problem with 3 robots. [center] The three projections of this problem's coordination space. The initial state is the lower-left corner of each; the goal is in the upper right. An example left-greedy path is shown. [right] The full 3-d coordination space.

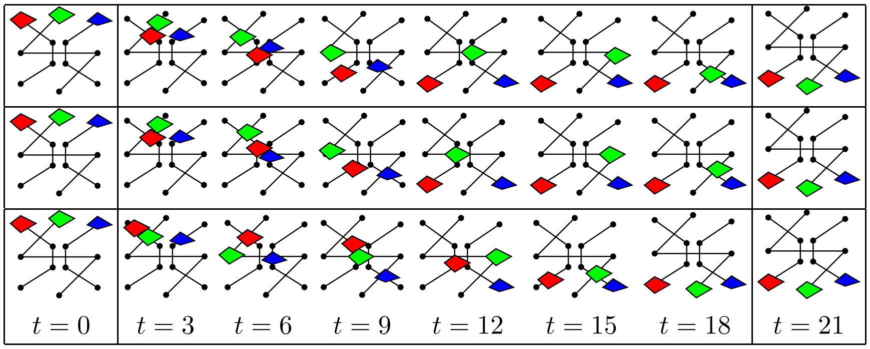

FIGURE 6. Workspace snapshots of the three Pareto optima for the problem in Figure 5.

4.2. **Dealing with nonmonotonicity.** We now relax the monotonicity requirement. The primary difficulty to be overcome is that obstacle orientations are no longer sufficient to determine a path's homotopy class. Indeed, one may revisit obstacles numerous times in opposite orientations.

To collate this data, extend rays upward and downward from each obstacle vertex that is a horizontal local extremum and do the same *mutatis mutandis* for vertical local extrema. For each ray r_i, assign a symbol a_i to represent the action of crossing this ray in the forward direction and let a_i^{-1} denote crossing the ray backward. The homotopy

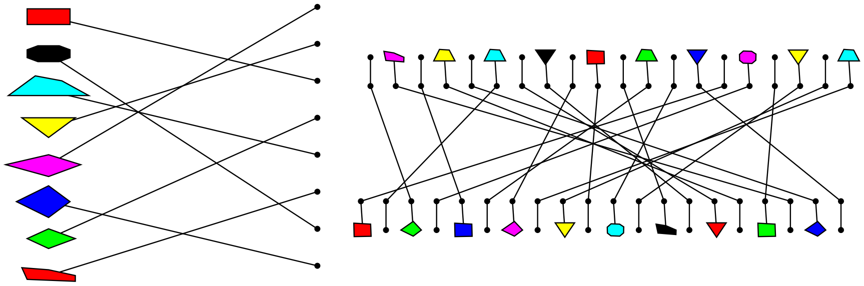

FIGURE 7. Two complex coordination problems; [left] for 8 robots, the 37 optimal coordinations were computed in 171.6 seconds by an exhaustive search of monotone classes; [right] for 20 robots, a single coordination was computed in 1.53 seconds.

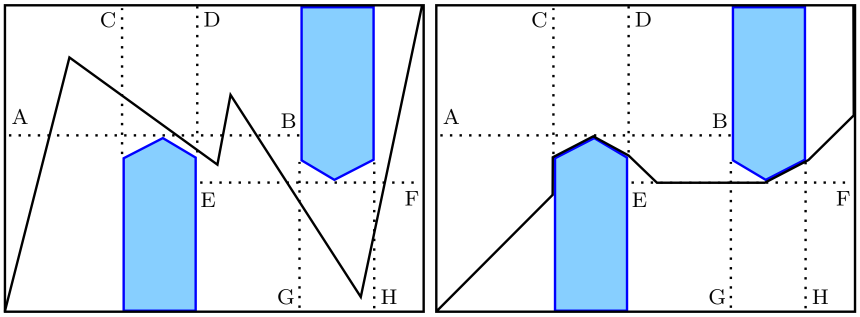

FIGURE 8. A 2-d coordination space in which a nonmonotonic solution path is needed. Dashed lines indicate the critical events. [left] A sample path with crossing sequence $c_1 = ACB^{-1}DBB^{-1}E^{-1}GHF$. [right] The optimized path has crossing sequence $c_2 = CAB^{-1}DE^{-1}GFH$. In the group L, c_1 and c_2 are equivalent.

class of a path is completely determined by the sequence in which it crosses these critical events (extending our earlier usage of the term) and the direction of each crossing. That is, a path's homotopy class is uniquely determined by its representation in the language $L = \{a_i, a_i^{-1}\}^*$. Conversely, if two paths are homotopic, then by continuously deforming one into the other, the crossing sequence will change by

- a transposition when the deformation passes an intersection point between a pair of critical events,
- a transposition when the order of successive events in unrelated projections is reversed, or
- cancellation of a crossing with its inverse.

Thus, we treat inverses in the usual way and admit commutativity between symbols corresponding to events in unrelated projections and between intersecting events in the same projection. We consider L with these relations as a group under word concatenation.

The length of the strings representing a homotopy class of a path can vary if there are "extraneous" elements that can be eliminated by commuting one or more pairs of symbols and canceling a symbol with its inverse. Since any extraneous crossings represent extra path length and so can only increase overall cost, any path that is cost-optimal up to homotopy will have a minimal length crossing sequence within its homotopy class. Such a minimal sequence dictates for each robot the order in which critical events should be crossed in a left-greedy path.

Given the crossing sequence for γ, we want to compute its minimal-length equivalent sequence. This can be accomplished using a simple algorithm that partitions the word by left-to-right sweep into maximal substrings in which each symbol commutes with each other. We call each such substring a **syllable**. Since $\mathcal{X}$ has dimension N, any mutual intersection of orthogonal hyperplanes will have degree at most N; hence the length of each syllable is bounded above by N. The word is minimized by shifting symbols to the left and canceling each symbol with its inverse when they appear in the same syllable. Details appear in Algorithm 2. To analyze this algorithm, let L denote the length of the word. Each iteration of the loop on lines 15-25 cancels at least one pair of symbols, so that loop executes $O(L)$ times. Each iteration makes $O(N^2)$ comparisons for each of the $O(L)$ pairs of adjacent syllables. As a consequence, the run time of the algorithm is quadratic in L. The word length L has complexity $\Theta(pm)$, in which p is the complexity of γ and m is the total complexity of the obstacle regions.

In light of this analysis, it is a simple matter to extend Algorithm 1 to optimize non-monotone paths. Having computed the minimal crossing sequence from γ, we can build γ' segment-by-segment, enforcing the sequence of critical event crossings prescribed for each robot. Each robot should stop rather than cross any event other than the next prescribed for it in the sequence. At each step, the correct direction for each robot (*i.e.*, forward or backward) is determined by the direction required to reach the next event for that robot in the sequence. Algorithm 3 describes this process more precisely. Aside from the additional preprocessing time required for word reduction in L, the complexity is unchanged.

4.3. **Complexity of enumeration.** Coordination of large numbers of robots under tight constraints is computationally challenging.

Example 4.2. Consider the generalization of Example 2.1 with $\Gamma_i = [-2, 2]$ for $i = 1...N$ and having $\binom{N}{2}$ obstacles, each of the form $x_i^2 + x_j^2 < 1$ for $i < j$. The two Pareto optimal classes from Example 2.1 generalize (by the cubical symmetry of the space) to yield 2^{N-1} optimal classes between opposite corners in this coordination space.

Algorithm 2 MINIMIZEWORD$(a_1 a_2 \cdots a_l)$

```
{Divide the word into syllables.}
$i \leftarrow 1$
$L \leftarrow 1$
for $i \leftarrow 1 \ldots l$ do
   if COMMUTESWITHALL$(a_i, S_L)$ then
      $S_L \leftarrow$ APPEND$(S_L, a_i)$
   else
      $L \leftarrow L + 1$
      $S_L \leftarrow a_i$
   end if
end for

{Shift left to cancel until the word is minimized.}
$p \leftarrow$ TRUE
while $p$ do
   $p \leftarrow$ FALSE
   for $i \leftarrow 1 \ldots L$ do
      for each $a$ in $S_i$ do
         if COMMUTESWITHALL$(a, S_{i-1})$ then
            $(S_i, S_{i-1}) \leftarrow$ LEFTSHIFTANDCANCEL$(a, S_i, S_{i-1})$
            $p \leftarrow$ TRUE
         end if
      end for
   end for
end while
return CONCATENATE$(S_1, S_2, \ldots, S_L)$
```

This general type of obstacle set – N intersecting cylindrical regions – can be realized physically by an AGV system with N disc-like robots sliding along intervals, the collection of which have an N-fold intersection at their midpoints. In the example depicted in Figure 9[left], each $\mathcal{X}_{i,j}$ contains a single elliptical obstacle. The exponential complexity of this coordination problem is intuitively clear: since everyone cannot pass through the shared center at the same time, they must proceed through one at a time. There are a factorial number of ways so to do.

Example 4.3. Consider a problem with $N = 2n$ robots formed by combining n independent copies of the system in Example 2.1. Each of these subsystems generates 2 Pareto optimal solutions; the combined system has $2^{N/2}$ Pareto optimal solutions.

Example 4.3 is depicted in Figure 9[right].

Algorithm 3 ExactLeftGreedyNonmonotone($\mathcal{X}, \gamma$)

```
Require: $\gamma$ is a collision-free coordination in $\mathcal{X}$.
$x_0 \leftarrow x_{init}$
$W \leftarrow \emptyset$
$\mathcal{O} \leftarrow$ ObstacleRegions($\mathcal{X}$)
$E \leftarrow$ ExtendedCriticalEvents($\mathcal{O}, \gamma$)
$c_\gamma \leftarrow$ CrossingSequence($E, \gamma$)
$c_{\gamma'} \leftarrow$ MinimizeWord($c_\gamma, E$)
$t \leftarrow 0$
while $x_t \neq x_{goal}$ do
    $t \leftarrow t + 1$
    $v \leftarrow$ MaximalVelocityNonmonotone($x, W, \mathcal{O}, c_{\gamma'}$)
    $e \leftarrow$ NextEvent($x, v, R, E, \mathcal{O}$)
    $W \leftarrow$ UpdateWaitForSet($p, e, W$)
    $x_t \leftarrow x_{t-1} + v \cdot e$.time
end while
return $(x_0, \ldots, x_t)$
```

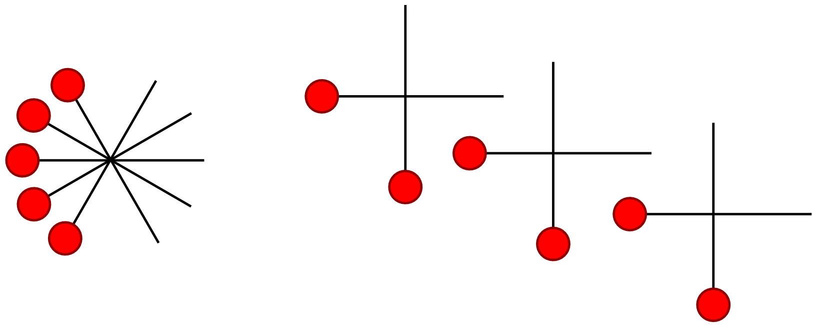

FIGURE 9. Two coordination problems for AGV-like systems. For each, the number of Pareto optimal solutions is exponential in the number of robots.

5. Discussion

5.1. **Conclusions and extensions.** It comes as no surprise that the enumeration of all Pareto optima is of high complexity: coordination is inherently difficult, and Lemma 1.1 implies that such an enumeration is akin to determining all minima for all monotone scalarizations of the cost functions. The algorithms we present are effective enough to be useful in factory AGV systems, where in practice the tracks have relatively low degree of intersection.

Although our examples and simulations are for convex PL robots sliding by rigid translation in the plane, our setting is much more general. In particular, our coordination spaces are for arbitrary roadmaps (not merely paths) in arbitrary C-spaces. Hence, one can have fully 3-d robots with internal degrees of freedom, nonholonomic constraints, or whatever one wishes, so long as the precomputed roadmap is one-dimensional.

Additional extensions of this work are possible. For example, [16] gives a proof of the finiteness bound and uniqueness result for more general roadmap coordination systems having moving workspace obstacles. Briefly, given a time-parameterized set of obstacles which move through the workspace, one builds an augmented coordination space which includes the parameterization variable (time) as an additional spatial variable. Since the methods we employ are topological in nature, they adapt very well to this setting. This is very robust: there are no assumptions on the moving obstacles' shapes or speeds.

A very worthwhile open problem is to prove the uniqueness and finiteness bounds for roadmap coordination systems which include acceleration bounds. Note that we have both assumed and made generous use of the fact that robots can start and stop instantaneously. Replacing this with acceleration bounds (both positive and negative) can invalidate many (and in some cases all) left-greedy paths. As such, it is highly nontrivial to classify the Pareto optimal equivalence classes for these systems. We believe but have no proof that the finiteness bounds still hold in the bounded acceleration case.

5.2. **Curvature: easy versus hard.** The entire program of classifying and computing Pareto optima hinges on the fact that left-greedy paths are unique up to homotopy on a cylindrical coordination space. This result, however, rests on the deeper foundation of CAT(0) geometry and the theorem that such cylindrical coordination spaces are nonpositively curved. As Examples 2.1 and 2.2 forcefully demonstrate, the difference between spaces with positive curvature and those without form the boundary between "easy" and "hard" classification problems for Pareto optima.

This is but one example of a truth broadly acknowledged in geometry, topology, and geometric group theory: hard problems become simpler upon restricting to systems without positive curvature. For example, it is possible to associate to a an arbitrary algebraic group a space whose geometry and topology "mirrors" certain algebraic properties of the group (similar in spirit to the manner in which C-space geometry mirrors the robotic system it represents). One can therefore consider groups which are devoid of positive curvature: see, e.g., the seminal work of Gromov [17]. It is well known that certain problem which are formally unsolvable for arbitrary groups (e.g., the *word problem*) become solvable when restricted to the subclass of groups whose spaces are nonpositively curved. Indeed, several algebraic problems experience a significant drop in complexity when restricted to hyperbolic groups [14].

We envision a similar **cat(0) principle** in robotics problems: *difficult problems should experience a drop in complexity when restricted to systems with an underlying* CAT(0) *geometry*. This has already been observed in shape-planning problems for metamorphic robots

[1], with further demonstration in the present work. It is likely, however, that a great many additional problems in robotics can benefit from this perspective. We outline two such instances for which progress seems likely.

Shortest paths. Consider the problem of motion planning via path planning in C-space. In many cases, optimal motion planning corresponds to the shortest path problem. It is a well-established fact that the shortest path problem on a compact PL Euclidean domain is "easy" is dimension two and "hard" in higher dimensions. Specifically, it is polynomial (in the complexity of the domain) in dimension two [20] and NP-hard in dimension three [9]. Perhaps not coincidentally, all compact two dimensional Euclidean domains are nonpositively curved. We further note that by paying careful attention to the constructions of Canny and Reif, it can be shown that the PL domains they construct in their NP hardness proof can be made to be simply connected; however, there is a great deal of positive curvature implicit in their constructions.

Recent work of Mitchell and Sharir [28] claims that one can achieve an NP-complete result within a class of spaces which are cylindrically deleted, and hence nonpositively curved, but which have a (large) fundamental group (as measured by the number of generators). In retrospect, this is perhaps not surprising: one can construct a symmetric 3-d space with many homotopy classes of shortest paths. By perturbing this space slightly, it becomes very difficult to tell which of the vestigial homotopy classes contains the shortest path. In [28], the authors also claim to show a poly-time algorithm for certain 3-d spaces which are CAT(0) (and some which are not).

It remains an open question whether the complexity of the shortest path problem always drops from NP-hard to P in the subclass of CAT(0) domains. The fact that there is a unique shortest path eliminates most of the known proofs of NP-hardness.

Pursuit-evasion. There is a large variety of problems involving a pursuer chasing an evader in a domain [5, 18, 22, 33, 39]. Among the many possibilities, an evader can have bounded or unbounded velocity, the pursuer can know or not know where the evader is, and the method of capture can be either line-of-sight visibility or physical intersection [capture]. Given a fixed set of rules and a fixed Euclidean domain, one wants to know the existence of a strategy for which the pursuer will always win regardless of the evader's strategy, with probabilistic wins being also of recent interest [23].

In various formulations of the pursuit problem, it is possible to win in (simply-connected) two dimensional domains but not always possible to win in three dimensional domains. In such cases, the CAT(0) principle suggests that restricting to higher dimensional domains which have a CAT(0) geometry would be the right category of domains on which to search for a pursuit strategy.

In both of these settings — shortest path and pursuit problems — the obstructions arise in dimension three. As such, attention is focused on problems fixed in these dimensions. If indeed it is the case that a CAT(0) restriction is the right generalization, then it holds hope that these problems will have computable solutions within this class for *arbitrary* dimensions. This is the case for the work of this paper, and we hope that it will be the case in other problems as well that the *curse of dimensionality* is mitigated by the CAT(0) principle.

References

[1] A. Abrams and R. Ghrist State complexes for metamorphic robot systems. *Intl. J. Robotics Research* 23(7,8):809–824, 2004.

[2] S. Akella and S. Hutchinson. Coordinating the motions of multiple robots with specified trajectories. In *Proc. IEEE Int. Conf. on Robot. and Autom.*, pages 624–631, 2002.

[3] M. D. Ardema and J. M. Skowronski. Dynamic game applied to coordination control of two arm robotic system. In R. P. Hämäläinen and H. K. Ehtamo, editors, *Differential Games - Developments in Modelling and Computation*, pages 118–130. Springer-Verlag, Berlin, 1991.

[4] J. Barraquand and J.-C. Latombe. Robot motion planning: A distributed representation approach. *Int. J. Robot. Res.*, 10(6):628–649, December 1991.

[5] T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory.* Academic Press, London, 1982.

[6] Z. Bien and J. Lee. A minimum-time trajectory planning method for two robots. *IEEE Trans. Robot. & Autom.*, 8(3):414–418, June 1992.

[7] M. Bridson and A. Haefliger, Metric Spaces of Nonpositive Curvature, Springer-Verlag, Berlin, 1999.

[8] S. J. Buckley. Fast motion planning for multiple moving robots. In *IEEE Int. Conf. Robot. & Autom.*, pages 322–326, 1989.

[9] J. Canny and J. Reif. Lower bounds for shortest path and related problems. In *Proc. 28th Ann. IEEE Symp. Found. Comp. Sci.*, 49–60, 1987.

[10] C. Chang, M. J. Chung, and B. H. Lee. Collision avoidance of two robot manipulators by minimum delay time. *IEEE Trans. Syst., Man, Cybern.*, 24(3):517–522, 1994.

[11] B. Chazelle, H. Edelsbrunner, M. Grigni, L. J. Guibas, J. Hershberger, M. Sharir, and J. Snoeyink. Ray shooting in polygons using geodesic triangulations. *Algorithmica*, 12:54–68, 1994.

[12] H. Chitsaz, J. M. O'Kane, and S. M. LaValle. Exact Pareto-optimal coordination of two translating polygonal robots on an acyclic roadmap. In *Proc. IEEE International Conference on Robotics and Automation*, 2004.

[13] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer, Berlin, 1997.

[14] D. Epstein et al. *Word Processing in Groups.* Jones & Bartlett Publishers, Boston MA, 1992.

[15] M. Erdmann and T. Lozano-Perez. On multiple moving objects. In *IEEE Int. Conf. Robot. & Autom.*, pages 1419–1424, 1986.

[16] R. Ghrist and S. M. LaValle. Nonpositive curvature and Pareto optimal motion planning. Preprint, `www.math.uiuc.edu/∼ghrist/pareto.pdf`

[17] M. Gromov. Hyperbolic groups. In *Essays in Group Theory* (ed. S. Gersten), Mathematical Sciences Research Institute Publications 8 (Springer, New York, 1987) 75–263.

[18] L. J. Guibas, J.-C. Latombe, S. M. LaValle, D. Lin, and R. Motwani. Visibility-based pursuit-evasion in a polygonal environment. *International Journal of Computational Geometry and Applications*, 9(5):471–494, 1999.

[19] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2001.

[20] J. Hershberger and S. Suri. An optimal algorithm for Euclidean shortest paths in the plane. *SIAM J. Comput.*, 28(6):2215-2256, 1999.

[21] H. Hu, M. Brady, and P. Probert. Coping with uncertainty in control and planning for a mobile robot. In *IEEE/RSJ Int. Workshop on Intelligent Robots and Systems*, pages 1025–1030, Osaka, Japan, November 1991.

[22] R. Isaacs. *Differential Games.* Wiley, New York, NY, 1965.

[23] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion with limited visibility. In *Proc. ACM-SIAM Symp. on Discrete Algorithms*, 1053–1063, 2004.

[24] K. Kant and S. W. Zucker. Toward efficient trajectory planning: the path-velocity decomposition. *Intl. J. Robotics Research* 5(3), 72–89, 1986.

[25] D.G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12(1):28–35, 1983.

[26] S. M. LaValle and S. A. Hutchinson. Path selection and coordination of multiple robots via Nash equilibria. In *Proc. 1994 IEEE Int'l Conf. Robot. & and Autom.*, pages 1847–1852, May 1994.

[27] S. M. LaValle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Trans. on Robotics and Automation*, 14(6):912–925, December 1998.

[28] J. Mitchell abd M. Sharir. New results on shortest paths in three dimensions. In *Proc. 20th ACM Symp. on Comp. Geom.*, 124–133, 2004.

[29] J. Miura and Y. Shirai. Planning of vision and motion for a mobile robot using a probabilistic model of uncertainty. In *IEEE/RSJ Int. Workshop on Intelligent Robots and Systems*, pages 403–408, Osaka, Japan, May 1991.

[30] G. A. Niblo and L. D. Reeves. The geometry of cube complexes and the complexity of their fundamental groups. *Topology*, 37(3):621–633, 1998.

[31] P. A. O'Donnell and T. Lozano-Pérez. Deadlock-free and collision-free coordination of two robot manipulators. In *IEEE Int. Conf. Robot. & Autom.*, pages 484–489, 1989.

[32] L. E. Parker. Cooperative motion control for multi-target observation. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, pages 1591–1598, 1998.

[33] T. D. Parsons. Pursuit-evasion in a graph. In Y. Alavi and D. R. Lick, editors, Theory and Application of Graphs, pages 426441. Springer- Verlag, Berlin, 1976.

[34] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. In *Algorithmic Foundations of Robotics V, STAR 7*, Springer-Verlag, 221–237, 2004.

[35] Y. Sawaragi, H. Nakayama, and T. Tanino. *Theory of Multiobjective Optimization*. Academic Press, New York, NY, 1985.

[36] J. T. Schwartz and M. Sharir. On the piano movers' problem: III. Coordinating the motion of several independent bodies. *Int. J. Robot. Res.*, 2(3):97–140, 1983.

[37] T. Simeon, S. Leroy, and J.-P. Laumond. Path coordination for multiple mobile robots: a resolution complete algorithm. *IEEE Trans. Robot. & Autom.*, 18(1), February 2002.

[38] S.-H. Suh and K. G. Shin. A variational dynamic programming approach to robot-path planning with a distance-safety criterion. *IEEE Trans. Robot. & Autom.*, 4(3):334–349, June 1988.

[39] I. Suzuki and M. Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM J. Computing*, 21(5):863888, October 1992.

[40] P. Svestka and M. H. Overmars. Coordinated motion planning for multiple car-like robots using probabilistic roadmaps. In *IEEE Int. Conf. Robot. & Autom.*, pages 1631–1636, 1995.

DEPARTMENTS OF MATHEMATICS [RG] AND COMPUTER SCIENCE [JO,SL], UNIVERSITY OF ILLINOIS, URBANA, IL 61801, USA

E-mail address: ghrist@math.uiuc.edu ; {jokane, lavalle}@cs.uiuc.edu