Safe Cooperative Robotic Patterns via Dynamics on Graphs

a paper delivered at the
Eighth International Symposium on Robotics Research
Shonan Village, Japan
October 3-6, 1997

Robert W. Ghrist*
Department of Mathematics
The University of Texas at Austin
Austin, TX 78712, USA

Daniel E. Koditschek†
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, MI 48109-2110, USA

Abstract

This paper explores the possibility of using vector fields to design and implement reactive schedules for safe cooperative robot patterns on graphs. The word "safe" means that obstacles — designated illegal portions of the configuration space — are avoided. The word "cooperative" connotes situations wherein physically distributed agents are collectively responsible for executing the schedule. The word "pattern" refers to tasks that cannot be encoded simply in terms of a point goal in the configuration space. The word "reactive" will be interpreted as requiring that the desired pattern be asymptotically stable: conditions close but slightly removed from those desired remain close and converge toward the desired pattern.

We consider Automated Guided Vehicles (AGV's) operating upon a predefined network of pathways, contrasting the simple cases of locally Euclidean configuration spaces with the more topologically intricate non-manifold cases. The focus of the present inquiry is the achievement of safe cooperative patterns by means of a succession of *edge point fields* combined with a *circulating field* to regularize collisions at non-manifold vertices.

1 Introduction

Recent literature suggests the growing awareness of a need for "reactive" scheduling wherein one desires not merely a single deployment of resources but a plan for successive re-deployments against a changing environment [24]. But scheduling problems have been traditionally solved by appeal to a discrete representation of the domain at hand. Thus the need for "tracking" changing goals introduces a conceptual dilemma: there is no obvious topology by which proximity to the target of a given deployment can be measured. In contrast to problems entailing the management of information alone, problems in many robotics and automation settings involve the management of *work* — the exchange of energy in the presence of geometric constraints. In these settings, it may be desireable to postpone the imposition of a discrete representation long enough to gain the benefit of the natural topology that accompanies the original domain.

This paper explores the possibility of using vector fields to design and implement reactive schedules for safe cooperative robot patterns on graphs. The word "safe" means that obstacles — designated illegal portions of the configuration space — are avoided. The word "cooperative" connotes situations wherein physically distributed agents are collectively responsible for executing the schedule. The word "pattern" refers to tasks that cannot be encoded simply in terms of a point goal in the configuration space. The word "reactive" will be interpreted as requiring that the desired pattern reject perturbations: conditions close but slightly removed from those desired remain close and, indeed, converge toward the exactly desired pattern.

*Supported by National Science Foundation Postdoctoral Fellowship DMS-9508846.
†Supported in part by National Science Foundation Grant IRI-9510673.

1.1 Motivation: Reactive Scheduling Suggests Feedback Controllers on Graphs

Graphs arise as a natural data structure by which to encode assembly plans, executive logic in machine tool cells, many other aspects of automation, and, indeed, many more diverse and abstract instances of plans and schedules. There is a well understood formal equivalence between graphs and finite automata [17], and the growing DES literature attests to the rich questions about controller synthesis that can arise in this context. The question further arises whether there might be merit to bringing scheduling problems into more intimate contact with the topology of continuous spaces wherein such notions as feedback and asymptotic stability have proven so useful in conferring robustness against unforseen perturbations. This paper explores a domain of scheduling problems whose setting is so tightly bound to the continuous world that the resolution of this question is motivated by more than purely intellectual speculation.

Graphs, and the desirability of imposing (or analyzing already imposed) dynamics upon them arise in robotic contexts in several different settings:

- As a representation of some designated cover of state space. The nodes of the graph correspond to the cells (subsets of the state space) in the cover, and connecting arcs denote non-empty intersection. Dynamics induced movement along the graph corresponds to the construction of vector fields whose flow brings an entire cell into into its intersection with the adjacent cell to be visited.

  For example, in [9], the second author and colleagues define a *prepares* relation — a partial order that prunes this graph into a tree — via controllers with point attractors in a specified neighborhood whose domain of attraction includes the entire cell (which is also positive invariant). We will in fact adopt the framework of that paper as a point of departure in the present work.

- When the state space admits the structure of a simplicial complex. The nodes of the graph correspond to the (varying dimensional) cells of the complex, and the arcs denote adjacency.

  For example, in robotic manipulations requiring successively higher order contact, it has proven possible to actually compute a topologically valid representation of the resulting simplicial complex in some cases [8]. Brockett [7] has introduced the graph described above and explored some aspects of dynamics imposed upon it.

- When the workspace is itself organized as a graph whose one dimensional "viaducts" connect workstation locations.

  For example, many factory materials handling systems are built using AGVs that must track a guidepath network embedded in the floor.

In all of these settings it makes sense to speak of patterns. In most of them are issues of safety, but typically passage to a graph presumes that the obstacles have been avoided by the manner in which the graph is embedded in the original configuration space. In some of these settings it makes sense to speak of multiple agents. In the last setting all of these considerations come together in a common problem, and the paper shall concentrate here in consequence.

1.2 Setting: AGV's on a Guidepath Network of Wires

An automated guided vehicle (AGV) is an unmanned powered cart "capable of following an external guidance signal to deliver a unit load from destination to destination" where, in most common applications, the guidepath signal is buried in the floor [10]. Thus, the AGV's workspace is a network of wires — a graph.[1] The motivation to choose AGV based materials handling systems over more conventional fixed conveyors rests not simply in their ease of reconfigurability but in the potential they offer for graceful response to perturbations in normal plant operation. In real production facilities, the flow of work in process fluctuates constantly in the face of unanticipated workstation downtime, variations in process rate, and, indeed, variations in materials transport and delivery rates [12]. Of course, realizing their potential robustness against these fluctuations in work flow remains an only partially fulfilled goal of contemporary AGV systems.

Choreographing the interacting routes of multiple AGVs in a non-conflicting manner presents a novel, complicated, and necessarily on-line planning problem. Nominal routes might be designed offline but they can never truly be traversed with the nominal timing, for all the reasons described above. Even under normal operating conditions, no single nominal schedule can suffice to coordinate the workflow as the production volume or product mix changes over time: new vehicles need to be added or deleted and the routing scheme adapted. In any case, abnormal conditions — unscheduled process down times; blocked work stations; failed vehicles — continually arise, demanding altered routes.

The traffic control schemes deployed in contemporary AGV systems are designed to simplify the real-time route planning and adaptation process by "blocking zone control" strategies. The workspace is partitioned into a small number of cells and, regardless of the details of their source and destination tasks, no two AGVs are ever allowed into the same cell at the same time [10]. Clearly, this simplification results in significant loss of a network's traffic capacity.

The contemporary robotic motion planning literature does not seem to offer much in the way of an al-

[1] This is not necessarily the case for beacon guided vehicles. However, their obvious advantages in flexibility and reconfigurability notwithstanding, their greater cost, complexity and relative fragility conspire to relegate their use to a small minority of installations [10].

ternative. Starting with pioneering work of Alami [1] there has been a small literature on multiple coordinated robots, but almost all papers seem to be concerned with offline versions of the problem. Latombe, in his excellent monograph [15] distinguishes between "centralized" and "decoupled" approaches to this problem. In the latter case, motion planning proceeds using multiple copies of the configuration space within which to situate a set of non-interacting robot vehicles. For a recent example of what Latombe terms a "coordinated" view of the decoupled case, Svestka and Overmars [25] introduce a "supergraph" on which multiple vehicles can be stepped through their individually specified paths, and vehicle-vehicle collisions prohibited by detaining one or another vehicle. For a recent example of what Latombe terms a "prioritized" view of the decoupled case, Lee et al. [16] compute k-shortest paths for each vehicle's source-destination pair, and work their way down the list of prefences based upon a vehicle's priority. It should be clear that in neither of these approaches has the recourse to blocking zone control been eliminated. Thus, while the decentralized approach side steps the inevitable curse of dimensionality, passing to the underlying configuration space seems to be required if the rigidity and inefficiency of blocking zone control strategies is to be eliminated.

The Industrial Engineering AGV literature seems chiefly concerned with higher level issues of layout and capacity [20] or dispatching and more general scheduling [2]. One interesting approach to layout seeks to avoid the subsequent traffic control problem entirely by clustering pickup and delivery stops in decoupled single vehicle loops [4]. In general, modeling the real-time factory floor is challenging enough that even the most recent treatments of these higher level layout and dispatching problems seem to rely on simulation rather than analysis for understanding the implications of one or another policy [3]. Here, again, is an indication that a dynamical systems point of view might shed additional light. For recent years have witnessed increasingly successful efforts to characterize such ensemble properties as the "mean transit time" induced by a flow. Thus, it seems to us entirely possible that a dynamics based network traffic control strategy might yield more readily to statistical analysis than present practice affords.

In this paper, we will consider a centralized approach that employs dynamical systems theory to focus on real-time responsiveness and efficiency as opposed to computational complexity or average throughput. No doubt, beyond a certain maximum number of vehicles, the necessity to compute in the high dimensional configuration space will limit the applicability of any algorithms that arise. However, this point of view seems not to have been carefully explored in the literature. Indeed, we will sketch some ideas about how an approach that starts from the coupled version of the problem may lend sufficient insight to move back and forth between the individuals' and the group's configuration spaces even in real time. For the sake of concreteness we will work in the so-called "pickup and delivery" (as opposed to the "stop and go" [4]) paradigm of assembly or fabrication, and we will not be concerned with warehousing style AGV applications.

In this context, a "pattern" amounts to a repetitive route through the graph (the particular sequence of workstations on the factory floor that the AGV must service). We desire a feedback policy that causes the robot to return to this pattern no matter what temporary obstructions or dislocations it experiences. We next desire to introduce two (or more) robots into the same graph, and seek a means of "juggling" them together that interleaves their patterns in an asymptotically stable and safe manner. That is, given a collection of patterns and a collection of corresponding feedback laws that achieve them for a single robot, we are interested in modifying the individual control strategies as modestly as possible so that a collection of robots can achieve those patterns simultaneously on the same graph with the guarantee of no collisions.

1.3 Contributions of the Paper

The paper is organized as follows. In §2, we review fundamental facts about the topology of graphs, without which, objects such as vector fields and gradients make little sense. We use this information to define the class of *edge point fields* — locally defined dynamics that realize single letter patterns. These act collectively as a toolbox from which to build a hybrid controller for achieving arbitrary patterns with a single AGV. This represents a slight generalization of the scheme the second author and colleagues have proposed in [9].

In §3, we turn to the problem of introducing multiple AGV's in the context of graphs which are manifolds. In this simpler case, it is often, but not always, possible to interleave controllers for single AGV's into a safe controller on the product configuration space.

The problem of dynamics and control on non-manifold graphs is then considered in §4. We present a fairly detailed analysis of the configuration space for a pair of AGV's on a Y-shaped graph — the simplest nontrivial situation. Here, a clarification of the configuration space presentation leads easily to a gradient-style construction that brings all initial conditions of two robots on the graph to any desired pair of goal points while guaranteeing safety (i.e., no collisions along the way). The desire for a more decoupled controller — the hope of an "interleaving" of otherwise independent individual patterns — impels a revised approach to safe navigation leading to the construction of a vector field that enables the AGV's to "dance" about one other at a vertex.

The dynamical features of this *circulating field* are suggestive of future hybrid constructions that might allow two independent patterns to be safely interleaved. We speculate about the form such juggling algorithms might take in §5.

2 Notation and Background

2.1 Graph Topology

A *graph*, Γ, consists of a finite collection of 0-dimensional vertices $\mathcal{V}:=\{v_i\}_1^N$, and 1-dimensional edges $\mathcal{E}:=\{e_j\}_1^M$ assembled as follows. Each edge is homeomorphic to the closed interval $[0,1]$ attached to $\mathcal{V}$ along its boundary points $\{0\}$ and $\{1\}$ [2]. We place upon Γ the quotient topology given by the endpoint identifications [21]: Neighborhoods of a point in the interior of e_j are homeomorphic images of interval neighborhoods of the corresponding point in $[0,1]$, and neighborhoods of a vertex v_i consist of the union of homeomorphic images of half-open neighborhoods of the endpoints for all incident edges.

The configuration spaces we consider in §4 are self-products of graphs. The topology of $\Gamma \times \Gamma$ is easily understood in terms of the topology of Γ as follows [21]. Let $(x,y) \in \Gamma \times \Gamma$ denote an ordered pair in the product. Then any small neighborhood of (x,y) within $\Gamma \times \Gamma$ is the union of a collection of neighborhoods of the form $\mathcal{N}(u) \times \mathcal{N}(v)$, where $\mathcal{N}(\cdot)$ denotes neighborhood within Γ. In other words, the products of neighborhoods form a *basis* of neighborhoods in the product space.

Given a graph, Γ, outfitted with a finite number n of noncolliding AGV's constrained to move on Γ, the configuration space of safe motions is defined as

$$\mathcal{C}:=(\Gamma \times \ldots \times \Gamma) - \mathcal{N}(\Delta),$$

where $\Delta:=\{(x_i) \in \Gamma \times \ldots \times \Gamma : x_j = x_k \text{ for some } j \neq k\}$ denotes the pairwise diagonal and $\mathcal{N}(\cdot)$ denotes (small) neighborhood.

For general graphs, the topology of $\mathcal{C}$ can be extremely complicated, as measured by, say, the rank of the fundamental group (see [21] for definitions). Even in the case where the workspace, Γ, is contractible (and thus, the product of its n copies along with the associated diagonal are all contractible), removal of this collision set often creates spaces with large fundamental group.

For example, given a graph Γ_K with K edges all connected at a single point (forming an K-pronged "star"), we can show that the fundamental group of the configuration space $\Gamma_K \times \Gamma_K - \mathcal{N}(\Delta)$ is a free group on $K^2 - 3K + 1$ generators — i.e., the number of "independent" closed paths in this space (with respect to continuous deformation) grows quadratically with K.

We do not treat any general aspect of this problem comprehensively in this paper; rather, we restrict attention to several simple preparatory examples and one basic but nontrivial example which illustrates nicely the relevant features present in the more general situation.

In order to proceed, it is necessary to clarify what we mean by a vector field on a simplicial complex that fails to be a manifold. This is a nontrivial issue: for example, in the case of a graph, the tangent space to a vertex with incidence number greater than two is not well-defined. Clearly, graphs posessing such vertices are not manifolds. But every graph can be embedded in a Euclidean space with edges "pinched together" so that the tangent vectors at each vertex all lie within the same well-defined one-dimensional tangent space to the embedding [22]. The possible ambiguity in pairwise orientation between the tangent spaces to each edge endpoint is resolved by choice of some convention: here we will always assume that positive vectors are outward directed on the right boundary, 1, and inward directed on the left boundary, 0, of an edge. This yields an existence and (forward time) uniqueness guarantee for solutions to the differential equations defined in this manner [22].

For present purposes, we find it convenient to work with an intrinsic (i.e., directly in the graph rather than via an embedding in a Euclidean space) formulation of this property. To this end, denote by v a vertex with K incident edges $\{e_i\}_1^K$, and by $\{X_i\}_1^K$ a collection of nonsingular vector fields locally defined on a neighborhood of the endpoint of each e_i (homeomorphic to $[0,1)$). These meet the conditions for existence of solutions with respect to some "pinched" embedding of Γ if and only if (1) the magnitude of the endpoint vectors $\|X_i(0)\|$ (taken with respect to the attaching homeomorphisms) are all identical; and (2) the *signs* of the endpoint vectors $X_i(0)$ (either positive if pointing into $[0,1)$ or negative if pointing out) are not all the same. Heuristically, condition (1) implies that we may identify all the endpoints to the vertex while respecting the vector field, and condition (2) means that the negative directions flow into the vertex — the positive directions flow out. If there are only positive or only negative directions, we do not have a well-defined vector field at the vertex — i.e., the existence of solutions may not be guaranteed. .

Since graphs need not be manifolds, results such as uniqueness of solutions are not guaranteed, and, in fact, are not true in general. The best one can hope for is to have a vector field generating a *semiflow* — that is, a continuous dynamical system with unique forward orbits. Pursuing the intrinsic formulation of such a "forward uniqueness property, " observe that for a vector field on a graph, it is clear that a semiflow is defined if and only if there is a *unique* edge along which the vector field is positive. In other words, all orbits through the vertex emanate along a single edge. Note that running a semiflow in backwards time does not generate unique orbits, unless the semiflow was actually a flow.

These notions of intrinsic vector fields generating semiflows on graphs extend naturally to the cross products of graphs we consider later. Again, the theme is to embed such spaces smoothly into a higher dimensional Euclidean space inducing well-defined tangent space. For the remainder of this work, we will define all vector fields intrinsically, without worrying about the specific embedding required.

[2] We will assume away in the sequel the possibility of "homoclinic" edges whose boundary points are attached to the same vertex.

2.2 Edge Point Fields

In the context of describing and executing *patterns* or periodic motions on a graph, one desires a set of building blocks for moving from one goal to the next. We thus introduce the class of *edge point fields* as a dynamical toolbox for a hybrid controller. Given a specified goal point $g \in e_j$ within an edge of Γ, an *edge point field* is a locally defined vector field X_g on Γ with the following properties:

Locally Defined: X_g is defined on a neighborhood $\mathcal{N}(e_j)$ of the goal-edge e_j within the graph topology. Furthermore, forward orbits under X_g are uniquely defined.

Point Attractor: every forward orbit of X_g asymptotically approaches the unique fixed point $g \in e_j$ [3].

Navigation-Like : X_g admits a C^0 Lyapunov function, $\Phi_g : \Gamma \to R$.

Lemma 1 *Given any edge $e_j \subset \Gamma$ which is contractible within Γ, there exists an edge point field X_g for any desired goal $g \in e_j$.*

Proof: Fix the desired goal $g \in \Gamma$. By hypothesis, e_j (and thus a neighborhood $\mathcal{N}(e_j)$) is contractible; hence, given any point $x \in \mathcal{N}(e_j)$, there exists a *unique* one-to-one path from g to x in $\mathcal{N}(e_j)$. For, if there were two such paths, this would imply the existence of a one-to-one map of a circle into $\mathcal{N}(e_j)$, contradicting contractibility. Place any bounded metric d on Γ and define the function $\Phi : N(e_j) \to R$ via $\Phi(x)$ is equal to the d-length of the unique path from g to x. The properties of the metric then guarantee that Φ is a function whose gradient field is an edge point field for g, where we define the gradient to be the induced gradient on the interior of each edge under the homeomorphisms to $[0, 1]$ via the topology on Γ. At the vertex, the gradient is taken with respect to the unique edge along which the function descends. This is compatible with the given definitions for vector fields on graphs.

The only occasion for which an edge e_j is not contractible in Γ is in the "homoclinic case" when both endpoints of e_j are attached to the same vertex, forming a loop. In such instances, one may avoid the problem by subdividing the edge to include more vertices, which is very natural in the setting of this paper, where vertices correspond to workstations along a path.

2.3 Discrete Regulation of Patterns

We adopt the standard framework of symbolic dynamics [17]. By an excursion on a graph is meant a (possibly infinite) sequence of edges from the graph (including the empty edge), $E = e_{i_1} \dots e_{i_N} \dots \in \mathcal{E}^Z$, having the property that each pair of contiguous edges, e_{i_j} and $e_{i_{j+1}}$ share a vertex in common. The set of excursions forms a language, $\mathcal{L}$: the so-called *subshift* on the alphabet defined by the named edges (we assume each name is unique) [17]. The *shift operator*, σ, defines a discrete dynamical system on the set of excursions, mapping the set of infinite sequences into itself by decrementing the time index. An *M-block extension* of the original language arises in the obvious way from grouping together each successive block of M contiguous letters from an original sequence, and it is clear how σ induces a shift operator, σ^M on this derived set of sequences.

Given a legal block, $B = e_{i_1} \dots e_{i_M} \in \mathcal{L}$, we will say that an excursion realizes that pattern if its M-block extension eventually reaches the "goal" $BBBBB \dots$ under the iterates of σ^M. In other words, after some finite number of applications of σ, the excursion consists of repetitions of the block B (terminating possibly with the empty edge).

In a previous paper [9], the second author and colleagues introduced a very simple but effective discrete event controller for regulating patterns on graphs from all reachable initial edges by pruning the graph back to a tree (imposing an ordering). Of course, this simple idea has a much longer history. In robotics it was introduced in [18] as "pre-image backchaining;" pursued in [19] as a method for building verifiable hardened automation via the metaphor of a funneling; and in [11] as a means of prescribing sensor specifications from goals and action sets. In the discrete event systems literature an optimal version of this procedure has been introduced in [6] and a generalization recently has been proposed in [23].

Let $\mathcal{E}^0 := B \subset \mathcal{E}$ denote the edges of Γ that appear in the block of letters specifying the desired pattern. Denote by

$$\mathcal{E}^{n+1} \subset \mathcal{E} - \bigcup_{k \le n} \mathcal{E}^k$$

those edges that share a vertex with an edge in $\mathcal{E}^n$ but are not in any of the previously defined subsets. This yields a finite partition of $\mathcal{E}$ into "levels," $\{\mathcal{E}^p\}_{p=0}^P$, such that for each edge, $e_i^p \in \mathcal{E}^p$, there can be found a legal successor edge, $e_j^{p-1} \in \mathcal{E}^{p-1}$, such that $e_i^p e_j^{p-1} \in \mathcal{L}$ is a legal block in the language. Note that we have implicitly assumed $\mathcal{E}^0$ is reachable from the entire graph — otherwise, there will be some "leftover" component of $\mathcal{E}$ forming the last cell in the partition starting within which it is not possible to achieve the pattern. Note as well that we impose some ordering of each cell $\mathcal{E}^p = \{e_i^p\}_{i=1}^{M_p}$: the edges of $\mathcal{E}^0 = B$ are ordered by their appearance in the block; the ordering of edges in higher level cells is arbitrary.

We may now define a "graph controller" law, $G{:}\mathcal{E} \to \mathcal{E}$ as follows. From the nature of the partition $\{\mathcal{E}^p\}$ above, it is clear that the least legal successor function,

$$L(p,i) := \begin{cases} i+1 \bmod M & : p = 0 \\ \min\{j \le M_p : e_i^p e_j^{p-1} \in \mathcal{L}\} & : p > 0 \end{cases}, \quad (1)$$

[3] When it is not clear from the context, we shall denote the goal point achieved by an edge point flow as $\mathbf{g}(X_g) = \{g\}$.

is well-defined. From this, we construct the graph controller:

$$G(e_i^p) := e_{L(p,i)}^{p-1}. \tag{2}$$

It follows almost directly from the definition of this function that its successive application to any edge leads eventually to a repetition of the desired pattern:

Proposition 2 *The iterates of G on $\mathcal{E}$ achieve the pattern B.*

2.4 Hybrid Edge Point Fields

A semiflow, $(X)^t$, on the graph induces excursions in $\mathcal{L}$ parametrized by an initial condition as follows. The first letter corresponds to the edge in which the initial condition is located (initial conditions at vertices are assigned to the incident edge along which the semiflow points). The next letter is added to the sequence by motion through a vertex from one edge to the next.

We will say of two edge point fields, X_1, X_2 on a graph, Γ, that X_1 *prepares* X_2, denoted $X_1 \succ X_2$, if the goal of the first is in the domain of attraction of the second,

$$\mathrm{g}(X_1) \subset \mathcal{N}(X_2).$$

The prepares relation imposes a partial order over the edge point fields on Γ. Given any finite collection of edge point fields on Γ, we will choose some $0 < \alpha < 1$ and assume that their associated Lyapunov functions have been scaled in such a fashion that $X_1 \succ X_2$, implies

$$(\Phi_1)^{-1}[0, \alpha] \subset \mathcal{N}(X_2).$$

In other words, an α crossing of the trajectory $\Phi_1 \circ (X_1)^t$ signals arrival in $\mathcal{N}(X_2)$.

Suppose now that for every edge in some pattern block, $e_i^0 \in \mathcal{E}^0$, there has been designated a goal point, g_i^0, along with an edge point field X_i^0 taking that goal, $\mathrm{g}(X_i^0) = g_i^0$. Assume as well that the edge point field associated with each previous edge in the pattern prepares the flow associated with the next edge, in other words, using the successor function (1) we have,

$$\mathrm{g}\left(X_j^0\right) \subset \mathcal{N}\left(X_{L(j)}^0\right).$$

Now construct edge point fields on all the edges of Γ such that the tree representation of their $\succ$ relations is exactly the tree pruned from the original graph above — namely we have

$$\mathrm{g}\left(X_j^p\right) \subset \mathcal{N}\left(X_{L(j)}^{p-1}\right).$$

We are finally in a position to construct a hybrid semi-flow on Γ. This feedback controller will run the piece-wise continuous vector field, $\dot{x} = X$, as follows

$$X := \begin{cases} X_j^p & : x \in e_j^p \text{ and } \Phi_j^p > \alpha \\ X_{L(j)}^{p-1} & : x \in e_{L(j)}^{p-1} \text{ or } \Phi_j^p \leq \alpha \end{cases}. \tag{3}$$

It is clear from the construction that progress from edge to edge of the state of this flow echoes the graph transition rule G, constructed above.

Proposition 3 *The edge transitions induced by the hybrid controller (3) are precisely the iterates of the graph map, G, (2) in the language, $\mathcal{L}$.*

3 Single Letter Patterns on Manifolds

In the voluminous literature concerned with artificial potential fields for robotic path planning, one often encounters the notion of a "locally repelling" field — a C^∞ (and typically piecewise analytic) gradient field capable of repelling in the neighborhood of an obstacle but otherwise (i.e., away from that neighborhood) completely goal-minded. In the context of multiple robots inhabiting the same workspace, this motivates the notion of "interleaving" controllers that promote the goals of each individual robot in isolation most of the time, reacting (in an appropriately repelling manner) to the presence of others only in configurations near the diagonal. We suspect that there may be fundamental obstructions to such strategies in configuration spaces with sufficiently large fundamental group.

In this section we explore the notion of "interleaving" in the simplest setting: where the goal of each robot consists of a single point and where the workspace is a manifold. Even here, it becomes apparent that there may exist intrinsic obstructions to matching up fully decoupled goal-seeking fields with repelling fields in the neighborhood of the diagonal.

3.1 The Line Segment

A segmented line, Λ, is any graph homeomorphic to $\mathcal{X} = [-1, 1]$ — for example, any connected graph with one edge and two vertices.

Now suppose $g \in \Lambda$ is a point goal and let $h_g : \Lambda \to \mathcal{X}$ be a diffeomorphism that takes the two endpoints of Λ to $\{-1\}, \{1\} \subset \mathcal{X}$, respectively, and takes g to the origin, $0 \in \mathcal{X}$. Then

$$\gamma_g(x) := (h_g(x))^2/2 \tag{4}$$

is a navigation function for g on Λ according to the definition in [14].

We now show how to build a safe cooperative version of this pattern with two robots. Define $\delta(x) := |x_1 - x_2|^2$. Since the diagonal,

$$\Delta := \delta^{-1}[0] \subset \Lambda \times \Lambda \approx \mathcal{X} \times \mathcal{X},$$

disconnects the configuration space, it suffices to consider cooperation with two AGV's. [4] Consider, then, the case of a pair of AGV's within a connected component $\mathcal{C}_0$ of the configuration space,

$$x := (x_1, x_2) \in \mathcal{C}_0.$$

[4] Note, however, looking ahead to a discussion to follow, that for m points on a line segment, the configuration space has $(m-1)!$ path components, given by the ordering of the points on the line.

Assume that there is some goal $g = (g_1, g_2) \in \mathcal{C}_0$. We have

$$\gamma_g(x) := \gamma_{g_1}(x_1) + \gamma_{g_2}(x_2),$$

as a safe navigation function for (g_1, g_2) on the self cross product $\mathcal{X} \times \mathcal{X}$, but not safe, in general, on the diagonally severed component, $\mathcal{C}_0$.

It is established in [13] that

$$\Phi(x) := \gamma_g(x)/\delta(x)$$

is a safe navigation function for $g = (g_1, g_2) \in \mathcal{C}_0$. Thus, the gradient vector field associated with Φ produces the safe pattern defined by the point goal g.

One might have wished, instead, for a " more decoupled" controller than one arising from the vector field $grad\,\Phi$. For example, variants on the scaled cross product vector field

$$(grad\,(\gamma_{g_1}/\delta)\,, grad\,(\gamma_{g_2}/\delta))$$

appear to yield safe navigation functions on $\mathcal{C}_0$ as well [5]. Notice that such a construction still presumes that centralized information regarding the state of each agent is available to all of them but relaxes their need to share information about their individual goals. Moreover, this formula might lend itself nicely to a C^∞ "blending" with the fully decoupled gradient field $grad g_1 \times grad g_2$ away from the boundary.

3.2 The Circle

Matters are much less satisfactory on the circle S^1 — a manifold which is not contractible. To begin with, every smooth edge point field will incur an unstable fixed point — only *essential* global attraction is possible. For example, identify S^1 with the unit circle in the complex plane $S^1 = \{e^{i\theta} : \theta \in [0, 2\pi)\}$. Then, given a goal θ_g, define the navigation function

$$\gamma_g(\theta) := 1 - \cos(\theta - \theta_g). \tag{5}$$

Since every continuous function on a compact set takes both a maximum and a minumum on that set, this construction (and, indeed, any other) necessarily introduces a spurious unstable fixed point, in this case, at $\theta_u := \theta_g - \pi$.

For understanding the configuration space of several AGV's on a circle, the following lemma (whose simple proof we omit) is key.

Lemma 4 *The configuration space $\mathcal{C}_{C,n}$ of n points on a circle is homeomorphic to $S^1 \times \mathcal{C}_{L,n-1}$, where $\mathcal{C}_{L,m}$ denotes the configuration space of m points on the line segment.*

From the previous subsection, we know that $\mathcal{C}_{L,m}$ is disconnected for $m > 1$. Since taking a cross product with S^1 does not change the number of connected components, we have that $S^1 \times S^1$ is not disconnected by the pairwise diagonal, but triple and higher cross products are.

We now explore the suggestion that the topology of the circle might preclude naive interleaving of individual navigation functions, in contrast to the situation for Λ. For example, consider the situation in which there are two AGV's with goals g_1 and g_2. By construction, we have a pair of navigation functions,

$$\gamma_1(\theta) := 1 - \cos(\theta - \theta_{g_1})\,;\ \gamma_2(\theta) := 1 - \cos(\theta - \theta_{g_2}).$$

One may interleave to form a navigation function

$$(\gamma_1(\theta_1) + \gamma_2(\theta_2))\,/\delta(\theta_1, \theta_2),$$

where δ is any nonegative function with $\delta^{-1}(0) = \Delta$, such as $1 - \cos(\theta_1 - \theta_2)$. Upon so doing, one can show that large regions of initial conditions do not reach the goal state. For example, in the case where $g_1 = 0$, $g_2 = \pi/2$, all initial conditions $\pi < \theta_1 < \theta_2 < 3\pi/2$ are trapped by the position of the sources and the diagonal repellor.

To construct safe navigation functions for attaining point goals of multiple AGV's on the circle, one may use the homeomorphism $h : \mathcal{C}_{C,n} \to S^1 \times \mathcal{C}_{L,n-1}$ of Lemma 4 to reduce to the cases already considered without the problems noted above. For example, in the case of two AGV's, the homeomorphism

$$h(\theta_1, \theta_2) := (\theta_1, \arg(\theta_2, \theta_1,)),$$

takes $\mathcal{C}_{C,2}$ to $S^1 \times (0, 2\pi)$. Here, we denote by $\arg(\theta_2, \theta_1,)$ the unique number $z \in (0, 2\pi)$ such that $\theta_1 + z = \theta_2$ modulo 2π. Given a pair of goals $\theta_{g_1} \neq \theta_{g_2}$, we have the corresponding point $h(\theta_{g_1}, \theta_{g_2}) = (\theta_{g_1}, x_{g_2})$, where $x_{g_2} := \arg(\theta_{g_2}, \theta_{g_1},)$. Construct the navigation function $\gamma_1(\theta)$ for θ_{g_1} as per (5). Then, construct the navigation function $\gamma_2(x)$ for x_{g_2} on the line segment $(0, 2\pi)$ using the obvious generalization of (4). Then, one obtains a safe navigation function Φ on $\mathcal{C}_{C,2}$ by pulling back the sum via the homeomorphism:

$$\Phi(\theta_1, \theta_2) := (\gamma_1 + \gamma_2) \circ h.$$

This may be extended to the case of three (or more) AGV's on the circle by using the product navigation functions for multiple points on the line segment via a similar homeomorphism h.

Unfortunately, the decoupled field $grad\,\gamma_1 + grad\,\gamma_2$ on the "model space", $S^1 \times \mathcal{C}_{L,n-1}$, is not at all decoupled when the pulled back to the physical setting in $\mathcal{C}_{C,n}$. We shall explore a similar phenomenon in the far more complicated case of the Y-graph.

The skeptical pragmatic reader may find motivation to persevere through our account of this next complication by noting that even the apparently contrived problem of scheduling AGVs on a single loop discussed in this section may, in itself, hold some practical value. For example, the choice of optimal dispatching and scheduling rules for multiple AGVs on a single loop seems to be far from understood within the Industrial Engineering community [2].

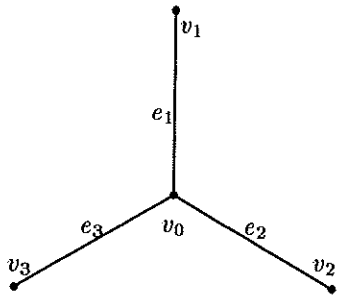

Figure 1: The Y-graph Υ.

4 The Y-Graph

In this section, we consider the simplest nontrivial example of a non-manifold configuration space: that associated to a *Y-graph*, Υ, having four vertices $\{v_i\}_0^3$ and three edges $\{e_i\}_1^3$, as illustrated in Figure 1. The topological features associated to such a system differ starkly from that of the previous examples. But this specical case has more general interest, since all graphs may be built up by gluing together K-pronged stars, of which Υ is the simplest nontrivial example. In any case, this setting certainly suggests the richness of the problem of dynamics on general graphs.

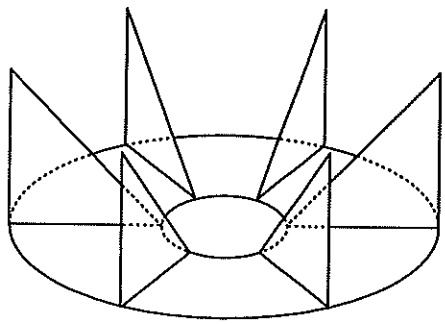

Figure 2: The configuration space $\mathcal{C}$ embedded in R^3.

Theorem 1 *The configuration space $\mathcal{C}$ associated to a pair of AGV's restricted to the Y-graph Υ is homeomorphic to an annulus with six 2-simplices attached as in Figure 2.*

Proof: The cross product $\Upsilon \times \Upsilon$ with the diagonal removed appears as in Figure 3: here, we have replaced every point of Υ with another copy of Υ and subsequently removed diagonal points, including the "center" of the graph. Note that this object as presented does not embed in R^3, but rather has an artificial self-intersection. To simplify the presentation, we deform the six "fins" created by the

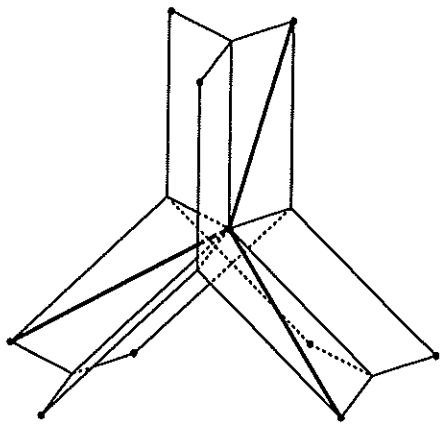

Figure 3: The space $\Upsilon \times \Upsilon - \Delta$.

diagonal cuts as in Figure 4, yielding the simplicial complex of Figure 5. This can be easily deformed into the hexagonal "star" of Figure 6, where the six radial lines correspond to where the diagonally severed fins were retracted. The center point, all that remains of the punctured diagonal set, has been removed. Upon reattaching these fins, we may transform the configuration space by a homeomorphism which takes the hexagonal star to a smooth annulus, yielding the final form of Figure 2, which does embed in R^3.

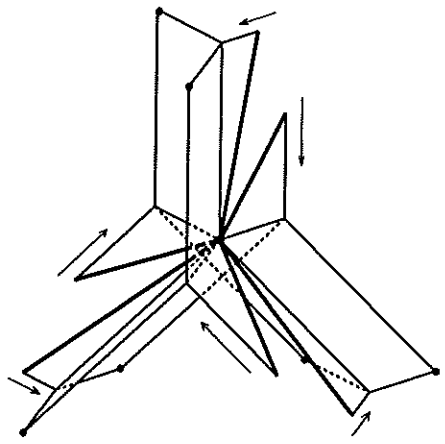

Figure 4: Retract the six fins of $\mathcal{C}$.

Corollary 5 *Given any pair of goals $g{:=}(g_1, g_2)$ where g_1 and g_2 live on different branches of Υ, there exists a navigation function (of class real-analytic off of the branch set) which sends all but a finite number of initial conditions to g under the gradient flow.*

Proof: On that portion of Figure 2 which is an annulus, the conditions for the theorems of

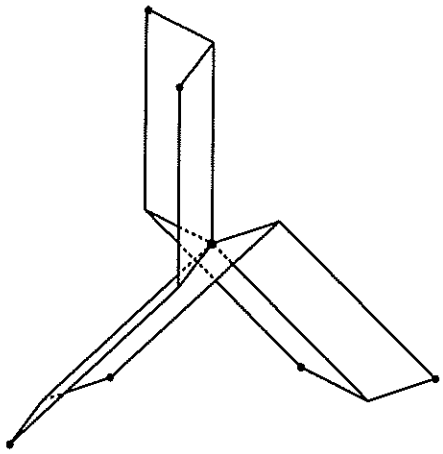

Figure 5: A simplicial complex with punctured center.

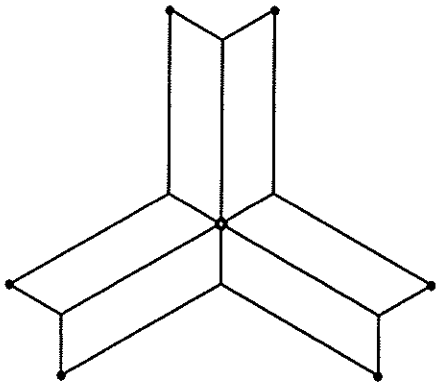

Figure 6: Flattening out yields a punctured hexagonal star.

Koditschek and Rimon [14] are met, since an annulus is a *sphereworld*. Hence, a navigation function on this subspace exists. Since the individual goals are not on the same branch of the graph, one may extend the navigation function to a function on the entire configuration space by defining the action of the flow on the fins to monotonically "descend" away from the diagonal and onto the annulus, where the implicitly defined function takes over as per our definitions for gradients and vector fields on non-manifolds. Note that upon prescribing the flow on the fins to send orbits onto the annulus, we have defined a semiflow, and hence have a well-defined navigational procedure.

This result is very satisfying in the sense that it guarantees a navigational function by applying existing theory to a situation which, from Figure 3 alone, would not appear to satify the conditions of being a sphereworld. However, it is not yet clear how such an application can be generalized to situations where the number of AGV's or the incidence number of the ambient graph increases. Hence, we consider an alternate solution to the problem of realizing compatible goals by means of a vector field on the configuration space. It is our belief that this method will readily adapt to complicated settings.

We construct a *circulating flow* on the configuration space $\mathcal{C}$ which has the effect of inducing a "dance" between the pair of AGV's until the appropriate configuration is reached.

Theorem 2 *There exists a piecewise-smooth vector field X on $\mathcal{C}$ which has the following properties:*

1. *X defines a nonsingular semiflow on $\mathcal{C}$;*
2. *The diagonal Δ is a repelling with respect to X; and*
3. *Every orbit of X approaches a unique attracting limit cycle on $\mathcal{C}$ which cycles through all possible ordered pairs of different edge-states.*

Proof: To construct the vector field, we introduce the following coordinate system fitted with respect to the topology of Υ. Let $\{e_i\}_1^3$ denote the three edges in Υ, parametrized so that $e_i \cong [0,1]$ with each $\{0\}$ identified at the center v_0 of Υ. Denote by $\hat{e}_i$ the unit tangent vector in each tangent space $T_x e_i$ pointing in the positive (outward) direction towards the endpoint v_i. Any point $x \in \Upsilon$ is thus given by a vector (x_1, x_2, x_3) in the $\{e_i\}$ frame, where $x_i \in [0,1]$ and at least two of these coordinates is zero. In other words, we are embedding Υ as the positive unit axis frame in R^3. Likewise, a point in $\mathcal{C}$ is given as a pair of distinct vectors (x, y), i.e., as a unit axis frame in $R^3 \times R^3 \cong R^6$ (see Figure 7). The reader should think of this as a collection of six unit coordinate planes, attached together pairwise along axes with the origin removed.

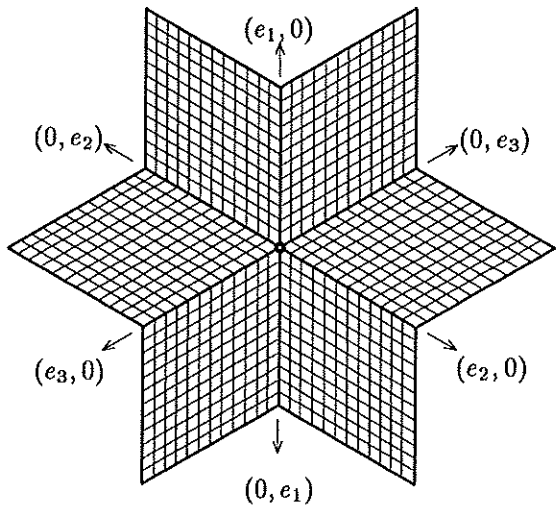

Figure 7: The coordinate system on the annular region of $\mathcal{C}$.

Any vector field on $\mathcal{C}$ may be uniquely represented as a pair of vectors in the $\{\hat{e}_i\}$ basis. Given a point $x \in \Upsilon$, denote by $\iota(x)$ the index of the nonzero coordinate of x (or by zero if $x = (0,0,0)$). Denote also by $|x|$ the value of the nonzero coordinate of x (or by zero if $x = (0,0,0)$). Thus, $x = |x|e_{\iota(x)}$. Any addition operation on the level of indices will always denote addition mod three.

The vector field we propose is the following: given $(x,y) \in \mathcal{C}$,

1. If $\iota(x) = \iota(y)$ then

$$\left.\begin{array}{l} \dot{x} = -|y|\hat{e}_{\iota(x)} \\ \dot{y} = 0 \end{array}\right\} \quad 0 < |x| < |y|$$
$$\left.\begin{array}{l} \dot{x} = 0 \\ \dot{y} = -|x|\hat{e}_{\iota(y)} \end{array}\right\} \quad 0 < |y| \leq |x| \qquad (6)$$

2. If $\iota(x) = \iota(y) + 1$ or $\iota(x) = 0$ then

$$\left.\begin{array}{l} \dot{x} = |y|\hat{e}_{(\iota(y)+1)} \\ \dot{y} = |y|(1-|y|)\hat{e}_{\iota(y)} \end{array}\right\} \quad 0 \leq |x| < |y|$$
$$\left.\begin{array}{l} \dot{x} = |x|(1-|x|)\hat{e}_{\iota(x)} \\ \dot{y} = -|x|\hat{e}_{\iota(y)} \end{array}\right\} \quad 0 < |y| \leq |x| \qquad (7)$$

3. If $\iota(y) = \iota(x) + 1$ or $\iota(y) = 0$ then

$$\left.\begin{array}{l} \dot{x} = -|y|\hat{e}_{\iota(x)} \\ \dot{y} = |y|(1-|y|)\hat{e}_{\iota(y)} \end{array}\right\} \quad 0 < |x| < |y|$$
$$\left.\begin{array}{l} \dot{x} = |x|(1-|x|)\hat{e}_{\iota(x)} \\ \dot{y} = |x|\hat{e}_{(\iota(x)+1)} \end{array}\right\} \quad 0 \leq |y| \leq |x| \qquad (8)$$

The vector field defines a nonsingular semiflow as follows: first, the vector field is by inspection nonsingular. Second, away from a neighborhood of the non-manifold points (where the fins attach to the annulus), the vector field is well-behaved and defines unique solution curves locally. Finally, at the non-manifold points, the vector field at any point is constructed so as to have two incoming directions and one outgoing direction; hence, the forward orbit of the vector field is uniquely determined.

This vector field admits a C^0 Lyapunov function $\Phi : \mathcal{C} \to [0,1)$ of the form

$$\Phi(x,y) := \left\{\begin{array}{ll} 1 - |(|x| - |y|)| & : \iota(x) = \iota(y) \\ 1 - \max\{|x|,|y|\} & : \iota(x) \neq \iota(y) \end{array}\right. . \qquad (9)$$

A simple calculation shows that on the fins ($\iota(x) = \iota(y)$), one has $\dot{\Phi} = -\max\{|x|,|y|\} < 0$, and furthermore that on the annulus ($\iota(x) \neq \iota(y)$), Φ changes as

$$\dot{\Phi}(x,y) = \Phi(\Phi - 1),$$

Hence, Φ strictly decreases off of the boundary of the annulus

$$L := \{(x,y) : |x| = 1 \text{ or } |y| = 1\} = \Phi^{-1}(0).$$

It follows from the computation of $\dot{\Phi}$ that the diagonal set Δ of $\Upsilon \times \Upsilon$ is repelling, and that the boundary cycle L is an attracting limit cycle.

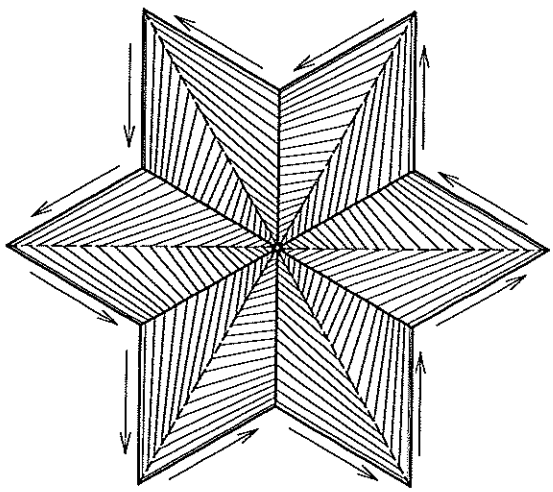

Figure 8: The circulating field on the annular region of $\mathcal{C}$.

The action of the vector field is to descend off of the "fins" of $\mathcal{C}$ onto the annular region, and then to circulate about the annulus while pushing out to the boundary cycle L, as in Figure 8. The Lyapunov function Φ decreases as the "height" when $\iota(x) = \iota(y)$ (on the fins) and decreases as the "radius" when $\iota(x) \neq \iota(y)$ (on the annulus). From this, it can be seen that the vector field constructed is merely the image of the vector field

$$\dot{r} = r(1-r) \ : \ \dot{\theta} = 1$$

under the homeomorphism taking the closed unit disc in the plane to the hexagonal star of Figure 7 by rescaling linearly along rays emanating from the origin. Of course, we may tune the radius of the limit cycle easily within this context and push it forward to a modified vector field on $\mathcal{C}$ by changing every term of the form $(1-|x|)$ and $(1-|y|)$ to, respectively, $(R-|x|)$ and $(R-|y|)$, where $0 < R < 1$ is the radius of the desired limit cycle.

5 Control of Dynamics on Graphs

The circulating vector field of §4 can be incorporated into a controller for dynamics on the Y-graph and many generalizations thereof. In this paper, we restrict ourselves to a brief sketch of how this may be accomplished, leaving details for a more comprehensive work.

The principal features of the vector field of §4 are as follows:

1. It generates a well-defined nonsingular semiflow on Υ.

2. Orbits of the semiflow attract onto an orbit which cycles through all ordered pairs of distinct edge combinations.

Consider the following simple hybrid controller on a general graph Γ with trivalent non-manifold points (i.e., there exists an atlas for Γ all of whose charts are homeomorphic copies of Υ). When the AGV's are geographically isolated (as measured by some natural metric d on each chart), use the edge point fields of §2 to send each AGV to its predetermined point goal, keeping track of the relative distances between AGV's. The distance between a pair of AGV's reaches a critical threshold when they are on a "collision course". Upon crossing the threshold, turn off the edge point fields, and turn on the circultaing field associated to the current chart on which the pair resides, using the chart homeomorphism to push forward the vector field of §4. The induced semiflow will by (1) and (2) above ensure that the AGVs' relative ordering on the troublesome edge are eventually permuted. When one of the AGV's has returned to its initial position with the prior obstruction removed, turn off the circulating field and resume the edge point field control.

In this way, the motion of the individual AGV's is to descend monotonically towards the goal except when a collision is imminent. Then, the circulating controller induces a "dance" between the pair which has the effect of moving the obstacle out of the way, allowing for further progress towards the goal. This prescribes an efficient, and courteous, traffic flow.

The future direction of this line of inquiry is clear: analogues of the circulating fields should be constructed for the more general cases of higher graph incidence and more AGV's. This presents a nontrivial topological problem, as noted earlier. Finally, it would be advantageous to implement a controller which is more modular — where incrementing the number of AGV's does not require a retooling of the entire controller. However, as noted in the case of dynamics on a circle, this is not always possible by straightforward interleaving of navigation functions.

Acknowledgements

We thank Professors Anthony Bloch, Philip Holmes and Stéphane Lafortune for a number of informative tutorial discussions bearing on the problems addressed in this paper.

References

[1] R. Alami, T. Simeon, and J.-P. Laumond. A geometrical approach to planning manipulation tasks. the case of discrete placements and grasps. In H. Miura and S. Arimoto, editors, *Robotics Research*, pages 453–463. MIT Press, 1990.

[2] J. J. Bartholdi and L. K. Platzman. Decentralized control of automated guided vehicles on a simple loop. *IIE Transactions*, 21(1):76–81, 1989.

[3] Y. A. Bozer and C. kuan Yen. Intelligent dispatching rules for trip-based material handling systems. *J. Manufacturing Systems*, 15(4):226–239, 1996.

[4] Y. A. Bozer and M. M. Srinivasan. Tandem configurations for automated guided vehicle systems and the analysis of single vehicle loops. *IIE Transactions*, 23(1):72–82, 1991.

[5] H. I. Bozma, C. S. Karagoz, and D. E. Koditschek. Assembly as a noncooperative game of its pieces: The case of endogenous disk assemblies. In *IEEE International Symposium on Assembly and Task Planning*, 1995.

[6] Y. Brave and M. Heymann. On optimal attraction of discrete-event processes. *Information Science*, 67(3):245–276, 1993.

[7] R. W. Brockett. Hybrid models for motion control systems. In H. L. Trentlman and J. C. Willems, editors, *Essays in Control: Perspectives in the Theory and Its Applications*. Birkhauser, Boston, MA, 1993.

[8] R. C. Brost. Computing metric and topological properties of configuration space obstacles. In *Proc. IEEE Conference on Robotics and Automation*, pages 170–176. IEEE, Scottsdale, AZ., 1989.

[9] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. *Int. J. Rob. Res.*, (to appear).

[10] G. A. Castleberry. *The AGV Handbook*. Braun-Brumfield, Ann Arbor, MI, 1991.

[11] M. Erdmann. Understanding action and sensing by designing actions-based sensors. *Int. J. Rob. Res.*, 14(5):483–509, 1995.

[12] S. B. Gershwin. *Manufacturing Systems Engineering*. Prentice Hall, Englewood Cliffs, NJ, 1994.

[13] D. E. Koditschek. An approach to autonomous robot assembly. *Robotica*, 12:137–155, 1994.

[14] D. E. Koditschek and E. Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11:412–442, 1990.

[15] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Press, Boston, MA, 1991.

[16] J.-H. Lee, B. H. Lee, M. H. Choi, J. D. Kim, K.-T. Joo, and H. Park. A real time traffic control scheme for a multiple agv system. In *IEEE Int. Conf. Rob. Aut.*, pages 1625–1630, Nagoya, Japan, 1995.

[17] D. Lind and B. Marcus. *Introduction to Symbolic Dynamics and Coding Theory*. Cambridge, 1995.

[18] T. Lozano-Perez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *Int. J. Rob. Res.*, 3(1):3–23, 1984.

[19] M. T. Mason. The mechanics of manipulation. In *Proc. International Conference on Robotics and Automation*, pages 544–548, March 1985.

[20] W. L. Maxwell and J. A. Muckstadt. Design of automatic guided vehicle systems. *IIE Transactions*, 14(2):114–124, 1981.

[21] J. R. Munkres. *Topology, A First Course.* Prentice Hall, 1975.

[22] D. Ruelle. *Elements of Differentiable Dynamics and Bifurcation Theory.* Academic Press, NY, 1989.

[23] R. Sengupta and S. Lafortune. An optimal control theory for discrete event control systems. *SIAM J. Control and Optimization*, (to appear).

[24] S. F. Smith. Reactive scheduling systems. In D. E. Brown and W. T. Schering, editors, *Intelligent Scheduling Systems*, pages 155–192. Kluwer Academic Publishers, Boston, MA, 1995.

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