Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

Composition of Local Potential Functions for Global Robot Control and Navigation

David C. Conner, Alfred A. Rizzi, and Howie Choset
{dcconner, arizzi,choset}@ri.cmu.edu
Carnegie Mellon University, Pittsburgh, PA 15213

***Abstract*— This paper develops a method of composing simple control policies, applicable over a limited region in a dynamical system's free space, such that the resulting composition completely solves the navigation and control problem for the given system operating in a constrained environment. The resulting control policy deployment induces a global control policy that brings the system to the goal, provided that there is a single connected component of the free space containing both the start and goal configurations. In this paper, control policies for both kinematic and simple dynamical systems are developed. This work assumes that the initial velocities are somewhat aligned with the desired velocity vector field. We conclude by offering an outline of an approach for accommodating arbitrary dynamical constraints and initial conditions.**

I. INTRODUCTION

The goal of the research discussed in this paper is to develop feedback control strategies that allow for automatic deployment of control policies in constrained environments, such that the resulting control system instantiates a natural, provably correct behavior for systems operating near their dynamic capabilities. Conventional robot architectures have separated the planning and control problem to a degree that provably correct planning algorithms offer no guarantees of dynamical performance. The typical approach is to operate far below the capabilities of the system in order to approximate kinematic behavior. By incorporating the constraints into low-level control policies, and planning in the space of available control policies, we offer a methodology that is provably complete in both kinematic and dynamical environments.

After introducing an overview of related work that inspire our approach, we discuss the decomposition and planning problem inherent in our work. This serves to motivate the type and scope of our low-level control policies, while offering proof of the completeness of the control policy composition. For clarity, we restrict the descriptions in this paper to $\mathbb{R}^2$, although the work directly extends to higher dimensions, as explained in the companion document [1]. Next we develop the low-level control policies, and provide proofs of the applicability to a simple dynamical system. We then outline the development of hybrid switching control policies required to overcome dynamical constraints, such as acceleration limits, and increase the domain of attraction for the low-level control policies. We conclude by outlining the next steps in our research plan, which is to develop automated systems of control policy deployment for real mobile robots subject to non-holonomic constraints.

Related Work. The problem of planning a path from start to goal has been extensively studied [2]. Unfortunately, just having a safe path does not guarantee that the robot can move safely. Constraints, in the form of kinematic and dynamical limitations on achievable velocities, may render the desired path impossible to track. Some heuristic methods, for example the *curvature-velocity method* [3], have been developed to encode dynamical limitations into the search space.

An early method to integrate planning and control was based on potential fields [4]. Unfortunately, most potential field constructions suffer from the well documented local minima problem [5], [6]. Connolly *et al.* [7] developed a potential field without local minima based on a numeric solution to Laplace's heat equation. Rimon and Koditschek [6] developed a local minima free potential field method for control by mapping a class of obstacles to a model space and generating a minimum free potential function over the model space; this potential function was termed a *navigation function*.

The approach we propose is based on decomposing the free space into cells, and solving the navigation and control problem for specific cells, basing control only on local information. The overall control policy is formed by composing the local control policies in a way that guarantees the overall performance. We take inspiration from the work of Burridge, Rizzi, and Koditschek [8], and the use of *sequential composition* to control a juggling robot. In their application, complex behaviors over a large domain were obtained by composing control policies designed to function over limited domains. Their work required hand-tuning and development of the specific control policy domains, and required the user to specify the deployment scheme. This paper also builds on prior work of the second author using sequential composition as a programming tool to specify robot motion programs for planar robots [9], [10].

The methods outlined in this paper make use of hybrid switching control policies; the control policies are switched as the system moves from cell to cell. For systems subject to dynamic constraints, switched control policies are also used within a cell to maximize the domain of savable states. Our control policies are defined such that the stability of the switched policies is guaranteed – this obviates the need for detailed analysis of the switching stability such as that of [11], [12]. The possibility of infinitely fast switching is precluded because the switching strategy presented here induces a partial order over the collection of control policies.

II. Overview

In this section we present an overview of a new approach to decomposing the problem of navigating a robot so that the planning and controls problem is solved simultaneously by inducing a global control policy that brings any initial configuration to the goal, provided that there is a single connected component of free-space containing both the start and goal configuration. This approach allows us to leverage the robustness afforded by feedback controls, while generating a globally convergent control policy.

A. Decomposition, Planning, and Control

Our approach to solving the robot navigation and control problems is to define a palette of control policies, and a switching strategy among the individual control policies, such that the resulting composition simultaneously solves both problems. We decompose the free space into a collection of cells, $\{\mathcal{P}\}$, defined as disjoint open sets. The union of the closures of the cells covers the free space, either exactly ($\mathcal{FS} = \cup\bar{\mathcal{P}}$) or to an approximation at some resolution($\mathcal{FS} \approx \cup\bar{\mathcal{P}}$) [2]. The collection of cells is referred to as a *cellular decomposition*. From the adjacency graph of the cells, with the root node corresponding to the cell containing the goal, we determine a partial ordering of the cells using a graph search algorithm, such as Dijkstra's algorithm [2].

With each cell, $\mathcal{P}$, we associate an individual control policy, which we term a *component control policy*. The component control policies are designed such that they will cause any configuration within the cell, $\mathcal{P}$, to move along a trajectory into a specified adjacent target cell, $\mathcal{P}_t$, specified by the partial order. During the evolution of the system trajectory, the configuration will not exit $\mathcal{P}$ other than by crossing the common boundary with $\mathcal{P}_t$. When the configuration enters the cell containing the goal configuration, a simple converging control policy (such as that presented in [9]) is used in place of the component control policy. Using the terminology of [8], we say that the component control policy associated with $\mathcal{P}$ *prepares* the component control policy associated with the target cell $\mathcal{P}_t$. Figure 1 shows an example of this technique.

The common boundary between a cell, $\mathcal{P}$, and the adjacent target cell, $\mathcal{P}_t$, is termed the *outlet zone* of $\mathcal{P}$. The boundary of a cell, excluding the outlet zone, is termed the *inlet zone*. Note, the outlet zone of $\mathcal{P}$ is part of the inlet zone of $\mathcal{P}_t$. If the goal configuration is contained in a cell, then the outlet zone is the empty set. The outlet zone for each cell is specified by the shared boundary with the adjacent cell next in the partial order, as determined by the adjacency graph.

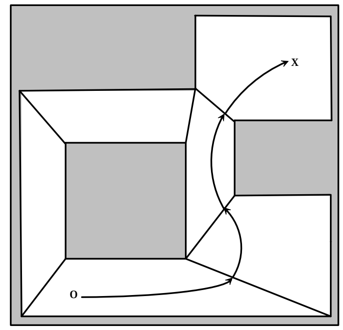

Fig. 1. A connected chain of cells determines a path from start to goal. Flow from previous cell prepares the control policy in the next cell. In this example, the free space was decomposed into convex polygons.

The specification of all of the outlet zones, and the resulting composition of the component control policies via the partial order, forms a hybrid control policy over the entire free space, which we term the *global control policy*. We refer to the specification of the control policies over the set of cells as a *control policy deployment*. The specified control policy deployment scheme is complete. In other words, given component control policies that function as specified in this section, the system can navigate a path from start to goal if and only if a chain of connected cells exists.

The control policy deployment outlined above depends on developing component control policies that generate the desired response for an arbitrary cell. Our approach is to develop a generic control policy that is valid for any cell in the set. In Section III, we discuss the development of such a control policy obtained by mapping the cell $\mathcal{P}$ to a simple model space, and generating a potential field over that model space. The cell is free of obstacles; therefore we can design a potential field over the model space that is free of local minima. The potential function is then *pulled back* onto the cell via a continuous mapping, φ. The gradient vector field of the *pullback* of the potential function induces a position dependent vector field over the cell $\mathcal{P}$. The vector field is suitable for control of kinematic systems. We generalize the kinematic control policy to yield a position dependent velocity reference control policy for dynamical systems.

B. Example Decomposition and Mapping

For our initial development, we chose a convex polygonal decomposition to approximate the free space of a system to some arbitrary resolution, as shown in Figure 1. The component control policy is designed to take the configuration through a designated *outlet face* into the adjoining polygon. There are a number of algorithms for decomposing spaces into convex polygons. For the demonstration in this paper, we assume the free space is represented as a general polygon and use an algorithm from Keil [13], allowing us to demonstrate automated deployment of our control policies.

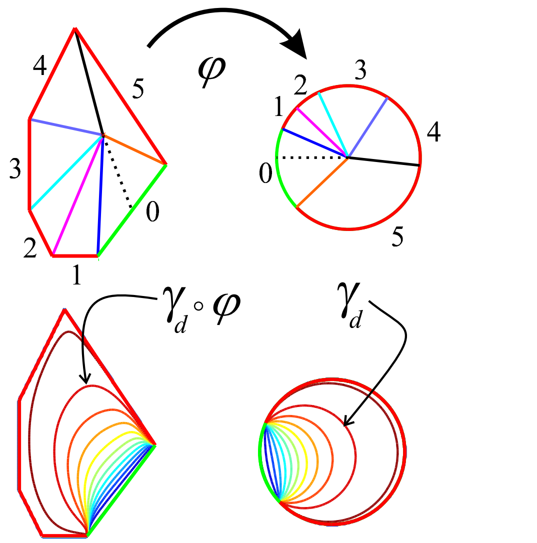

Fig. 2. Mapping from polygon to unit disk. The contour plot on the left shows level sets of the pullback $\gamma_d \circ \varphi$ on the polygon; the contour plot on the right shows the corresponding level sets of γ_d on the unit disk.

In order to generate a vector field with continuous derivatives, as needed by our dynamic control policy, we require a C^2 diffeomorphism from an arbitrary polygon to our model space. Unfortunately, the polygon vertices preclude the possibility of constructing a C^2 diffeomorphism. The mapping can only approximate the polygon at the corners while maintaining C^2 continuity. Our construction approximates the vertices in a natural way that averages the adjacent normals, thereby preserving the vector field properties presented in this paper. For details the reader is referred to [1].

We chose a unit disk for our model space, and define $\varphi : \mathcal{P} \to \mathcal{D}$, such that φ is a C^2 diffeomorphism from the interior of the polygon $\mathcal{P}$ to the interior of the unit disk,

$$\mathcal{D} = \left\{ (x_d, y_d) \mid r = \sqrt{x_d^2 + y_d^2} \leq 1 \right\} \subset \mathbb{R}^2 .$$

φ uses a radial retraction technique to map the cell boundaries to the boundary of the unit disk, using C^2 fillet curves[1] within a specified neighborhood of each vertex. This mapping is depicted in Figure 2.

III. Component Controllers

The control policy deployment outlined in Section II-A depends on developing component control policies that generate the desired response for each cell in the decomposition. In this section, we present a candidate component control policy for an arbitrary cell based on a potential field. Control Policies for both kinematic and dynamic systems are developed.

[1]In computer aided design (CAD), a *fillet curve* describes an arc joining two lines.

A. Harmonic Potential Functions

Our approach to generating a potential field, taking inspiration from [6], is to generate a potential function $\gamma_d : \mathcal{D} \to \mathbb{R}$, where $\mathcal{D}$ is our simple model space. The potential function in the arbitrary cell $\mathcal{P}$, given as the pullback of the model space potential, is

$$\gamma = \gamma_d \circ \varphi .$$

To generate a potential function free of local minima, we solve Laplace's equation

$$\nabla^2 \gamma_d = \frac{\partial^2 \gamma_d}{\partial x_d^2} + \frac{\partial^2 \gamma_d}{\partial y_d^2} = 0$$

over the unit disk. Laplace's equation, which is the classic partial differential heat equation, can be solved in closed form given the potential specification on the disk boundary, where the radius $r = 1$. Let $V(\theta)$ specify the potential along the boundary, where

$$V(\theta) = \begin{cases} V_b & \theta \in [\alpha_0, \alpha_1] \\ 0 & \text{otherwise} \end{cases} ,$$

with $\theta \in (-\pi, \pi]$, $\alpha_0 \in \left(-\pi, -\frac{\pi}{2}\right)$, and $\alpha_1 \in \left(\frac{\pi}{2}, \pi\right)$. The α_i terms are the angles of the mapped exit vertices in the disk. Specification of the boundary potential $(V_b > 0)$ provides a parametric freedom that can be used to control the potential field magnitude and resulting gradient magnitude. The closed form solution is given by

$$\gamma_d(r,\theta) = \frac{V_b(\alpha_1 - \alpha_0)}{2\pi} + \frac{V_b}{\pi}\left(\tan^{-1}\left(\frac{r\sin(\alpha_1-\theta)}{1-r\cos(\alpha_1-\theta)}\right) - \tan^{-1}\left(\frac{r\sin(\alpha_0-\theta)}{1-r\cos(\alpha_0-\theta)}\right) \right),$$

where $r = \sqrt{x_d^2 + y_d^2}$ and $\theta = \text{atan2}(y_d, x_d)$ is the polar coordinate representation of a point (x_d, y_d).

The potential function γ_d given by the solution of the heat equation is free of local minima or other critical points, and resembles a *navigation function* [6], [7]. The navigation function-like properties, including the absence of local minima, are preserved when γ_d is pulled-back onto the cell $\mathcal{P}$ via the diffeomorphism φ [6], [1]. Figure 2 shows an example of the pullback of a such a potential function.

We use the negative gradient of the pulled-back potential function to generate a vector field $\mathrm{X} : \mathcal{P} \to \mathcal{TP}$, given as

$$\mathrm{X}(\mathbf{q}) = -D_{\mathbf{q}}\gamma^T = -\left(D_{\mathbf{q}}\varphi^T D_{\mathbf{q}_d}\gamma_d^T\right)\big|_{\varphi(\mathbf{q})} , \qquad (1)$$

where $\mathbf{q} \in \mathcal{P}$ and $\mathbf{q}_d = \varphi(\mathbf{q}) \in \mathcal{D}$. By construction, the gradient vector field induced by the pullback of the heat equation solution is orthogonal to the boundary of the cell[2]. This is trivial to show, given that the potential along the boundary of the cell is constant by virtue of the pull-back. This orthogonality feature allows us to construct control policies that have continuity across the cell boundaries, and facilitates some of our later proofs.

[2]The vector field at the vertices of a polygonal cell of our example is undefined. In the neighborhood of the vertices, the mapping φ approximates the vector field in a natural way such that the desired behavior is preserved.

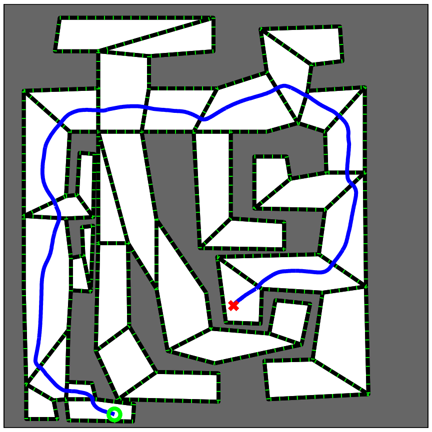

Fig. 3. Solution for kinematic system. The dark line shows the path taken from the start configuration, marked with an "o", to the goal configuration, marked with an "x". The dark region denotes the boundary of the free space, and dotted lines show the decomposition into convex polygons.

B. Kinematic Path Plan

For kinematic systems, the vector field $\mathrm{X}(\mathbf{q})$ can be used to specify the velocity of the system. The integral curves of $\mathrm{X}(\mathbf{q})$ trace a path from start to goal provided a connected set of cells exist between the start and goal.

Lemma 3.1: For a kinematic system of the form $\dot{\mathbf{q}} = \mathbf{u}$, where $\mathbf{q}, \mathbf{u} \in I\!R^2$, the integral curves of the vector field $\mathbf{u} = \mathrm{X}(\mathbf{q}) = -D_q\gamma^T$ for the pulled-back potential, determine a path from any point on the inlet zone, or interior of the cell, to a point on the exit zone such that the trajectory is completely contained in the cell until it crosses the exit zone.

The proof is trivial by construction.

Combined with the automated control policy deployment scheme described in Section II-A, the component control policy $\mathbf{u} = \mathrm{X}(\mathbf{q}) = -D_q\gamma^T$ yields a complete path planning method for the kinematic system via the induced overall control policy. Because the vector field is orthogonal to the boundaries, the velocity orientation is continuous. By normalizing the vector field, we can generate a continuous (except at the vertices) velocity profile from start to goal without changing the completeness properties of the algorithm.

Figure 3 shows an example deployment for a kinematic system using the convex polygon decomposition scheme from Section II-B. The free space is shown as a maze-like polygon, with a start and goal specified.

C. Dynamic Control Policy

While the control policy deployment scheme and component control policy design provides a novel solution to the classical mobile robot navigation problem, it does not guarantee that a real robot, subject to dynamical constraints, could follow the prescribed path.

Given a second order system of the form

$$\ddot{\mathbf{q}} = \mathbf{u}\,, \tag{2}$$

we wish to design a control law that converges to an integral curve of the vector field $\mathrm{X}(\mathbf{q}) = -D_{\mathbf{q}}\gamma^T$, without departing the defined cell $\mathcal{P}$. We begin by developing the control policy assuming the system allows arbitrary, but still finite, accelerations. Later, we consider acceleration and velocity constraints.

Using the negative gradient field as a position dependent velocity reference leads to a natural velocity reference control policy of the form

$$\mathbf{u} = K\,\left(\mathrm{X}(\mathbf{q}) - \dot{\mathbf{q}}\right) + \dot{\mathrm{X}}(\mathbf{q})\,, \tag{3}$$

where $K > 0$ is the "velocity regulation" gain [9]. The $K\,(\mathrm{X}(\mathbf{q}) - \dot{\mathbf{q}})$ term acts to decrease the error between the current and the desired velocity. The feed-forward term, $\dot{\mathrm{X}}(\mathbf{q}) = D_{\mathbf{q}}\mathrm{X}\,\dot{\mathbf{q}}$, accounts for the change in the vector field as we move in the $\dot{\mathbf{q}}$ direction, and allows the system to exactly track the integral curves of $\mathrm{X}(\mathbf{q})$ once the error has converged to zero.

Lemma 3.2: In the absence of constraints, including those of the cell boundary, the integral curves of the vector field $\mathrm{X}(\mathbf{q})$ are attractive to the trajectories of the closed loop system defined by (2) under the influence of (3).

Proof: For details, see [1]. The proof depends on a Lyapunov-like function using the velocity error $(\mathrm{X}(\mathbf{q}) - \dot{\mathbf{q}})$, and follows that given in [9]. ■

The orientation error, *i.e.* the angle between the desired velocity, $\mathrm{X}(\mathbf{q})$, and the current velocity, $\dot{\mathbf{q}}$, is given by

$$\vartheta = \cos^{-1}\frac{\dot{\mathbf{q}}^T\,\mathrm{X}}{\sqrt{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}}}\,,$$

where $\mathrm{X} = \mathrm{X}(\mathbf{q})$.

Lemma 3.3: In the absence of acceleration constraints, and for initial velocities such that $\dot{\mathbf{q}}^T\mathrm{X} > 0$, there exists a lower bound on K such that the orientation error, ϑ, monotonically decreases.

Proof: First, consider the isolated case where $\dot{\mathbf{q}} = 0$; we define $\vartheta\,|_{\dot{\mathbf{q}}=0} = 0$. Beginning from rest, the acceleration specified by $K\,(\mathrm{X}(\mathbf{q}) - \dot{\mathbf{q}})$ will move the system differentially in the direction of the desired velocity, and the orientation error will be zero. As we show below, the orientation error will remain zero for all time.

To show that the orientation error ϑ monotonically decreases, consider the set

$$\mathcal{U} := \{(\mathbf{q}, \dot{\mathbf{q}})\ \mid \vartheta = 0\}\,,$$

and define a Lyapunov-like function of the form

$$\begin{aligned}\eta_u &= \sin^2\vartheta = 1 - \cos^2\vartheta \\ &= \frac{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X} - \dot{\mathbf{q}}^T\mathrm{X}\,\dot{\mathbf{q}}^T\mathrm{X}}{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}}\,.\end{aligned} \tag{4}$$

Evaluating the time derivative of (4) along the trajectories of the closed loop system, and simplifying yields

$$\dot{\eta}_u = \frac{2\left(\dot{\mathbf{q}}^T\mathrm{X}\right)^2}{\left(\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}\right)^2}\Bigg[\bigg(\mathrm{X}^T\mathrm{X}\,\dot{\mathbf{q}}^T + \dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T - \frac{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}}{\dot{\mathbf{q}}^T\mathrm{X}}\dot{\mathbf{q}}^T - \frac{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}}{\dot{\mathbf{q}}^T\mathrm{X}}\mathrm{X}^T\bigg)\dot{\mathrm{X}} - K\left(\frac{\dot{\mathbf{q}}^T\dot{\mathbf{q}}\,\mathrm{X}^T\mathrm{X}}{\dot{\mathbf{q}}^T\mathrm{X}}\mathrm{X}^T\mathrm{X} - \mathrm{X}^T\mathrm{X}\,\dot{\mathbf{q}}^T\mathrm{X}\right)\Bigg] \quad (5)$$

When $\dot{\mathbf{q}}$ is aligned with $\mathrm{X}(\mathbf{q})$, $\dot{\eta}_u = 0$, and we conclude the set $\mathcal{U}$ is invariant. In other words, if the orientation error is zero, it remains zero.

Away from $\mathcal{U}$, the leading term of (5) is positive and bounded, because we assume that initially $\dot{\mathbf{q}}^T\mathrm{X} > 0$. Assuming the system has finite initial velocity and that $\|\mathrm{X}(\mathbf{q})\|$ is finite, if follows that the velocity error is finite; then by Lemma 3.2, the error magnitude decreases, and we conclude that $\|\dot{\mathbf{q}}\|$ remains finite for all time. The first parenthetical term in brackets for (5) has an indeterminate sign, but is finite since all the terms are bounded. The last parenthetical term in (5) is positive. Therefore, for sufficiently large K, $\dot{\eta}_u$ can be made negative definite. This implies that $\dot{\mathbf{q}}^T\mathrm{X}$ remains positive, and therefore $\dot{\eta}_u$ is always negative for sufficiently large K. Since $\dot{\eta}_u < 0$, we conclude that $\mathcal{U}$ is attractive and invariant, and that ϑ monotonically decreases under the influence of (3). ■

Intuitively, making K sufficiently large ensures that the control policy is correcting more quickly than the vector field is changing. The *sufficiently large* K is determined in the worst case based on the vector field derivative, which is largest near local concavities in the cell boundary.

Lemma 3.4: In the absence of acceleration constraints, with sufficiently large K and initial velocities such that $\dot{\mathbf{q}}^T\mathrm{X} > 0$, the trajectories of the closed loop system defined by (2) under the influence of (3), converge to the integral curves of the vector field $\mathrm{X}(\mathbf{q})$ in such a way that the trajectory never exits the cell except by the exit zone, and in fact exits the cell via the exit zone.

Proof: For $\dot{\mathbf{q}}^T\mathrm{X} > 0$, we know the orientation error is initially less than $\frac{\pi}{2}$. By Lemma 3.3, for sufficiently large K, the orientation error is monotonically decreasing.

Assume the trajectory exits the cell in the *inlet* zone. At the point of departure, $\dot{\mathbf{q}}^T\mathrm{X} < 0$ given the inward pointing vector field orthogonal to the cell boundary. This implies that $\vartheta > \frac{\pi}{2}$, requiring that the orientation error increased along its trajectory. This contradicts Lemma 3.3.

Since the vector field $\mathrm{X}(\mathbf{q})$ is nowhere zero over the cell, the system cannot come to rest and remain stationary, because the system experiences an acceleration along the vector field. Therefore, we conclude that the trajectory must leave the cell via the exit zone under the influence of (3) for the given conditions. ■

Figure 4 shows a simulation of the dynamic system given in (2) under the influence of (3). A variety of initial conditions are shown, each converging to the goal configuration using the hybrid control strategy induced by the underlying decomposition.

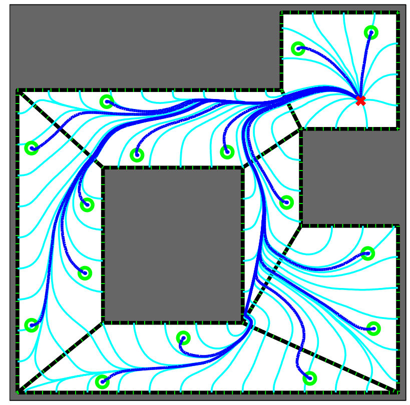

Fig. 4. Simulation of the dynamical system using the hybrid control strategy introduced in this paper. Light colored lines represent integral curves of $\mathrm{X}(\mathbf{q})$, while the dark colored lines represent trajectories of the system for various initial conditions. The velocity regulation gain used in this example was $K = 20$.

The utility of Lemmas 3.3 and 3.4 is limited by two factors. First, a large value for K can lead to an overly aggressive policy over the cell that may prove troublesome for implementation. Secondly, and most importantly, all real world systems have acceleration limits, which may very well be violated by the feed-forward term of (3), regardless of the value of K and the velocity error.

D. Constrained Dynamics

To extend our ideas to real world systems, we now consider the following dynamic constraints,

$$\|\dot{\mathbf{q}}\| \leq \mathrm{V}_{\max} \quad (6)$$
$$\|\mathbf{u}\| = \|\ddot{\mathbf{q}}\| \leq \mathrm{U}_{\max} \quad (7)$$

The velocity limit is taken to be a safety limit, and it is assumed that $\|\mathrm{X}(\mathbf{q})\| \leq \mathrm{V}_{\max}$ for all $\mathbf{q} \in \mathcal{P}$. However, if the change in the vector field $\mathrm{X}(\mathbf{q})$ is aligned with the current velocity, then the feed-forward term of (3) may act to increase the velocity magnitude (speed), causing a violation of the maximum speed.

The acceleration limit on the other hand, represents a physical limitation of the dynamic system. Since the acceleration cannot be exceeded, the inability to produce the desired acceleration will invalidate the lemmas given above. To accommodate both the velocity and acceleration constraints, we are currently developing modified control policies and hybrid switching strategies.

Our first proposed method for addressing the acceleration limits is by encoding a speed reduction in areas where the feed-forward term is excessive. By decreasing speed in areas where the vector field has high derivative changing, we reduce the impact of the feed-forward term, which depends on both the current velocity and vector field. For purposes of this paper, our only freedom for affecting the speed is in the specification of the boundary potential value, V_b. Unfortunately, this impacts the entire cell, so we are seeking methods based on the local magnitude of the $D_{\mathbf{q}}X$ term. This will allow us to naturally encode such common-sense notions as "slow down when approaching a sharp corner," based on local information (encoded in the decomposition) without explicitly modeling the corner.

The second approach we are pursuing, which is intended to address both acceleration and velocity constraints, is the development of hybrid switching control policies to form our component control policy. Our approach, again taking inspiration from the sequential composition methods of [8], [9], [10], is to define a set of control policies that are maximal over the state space of the cell $\mathcal{P}$, and whose composition converges to (3) before the system configuration exits the cell. The hybrid control policies are constructed such that they induce a partial order within a given cell, and preclude the possibility of infinitely fast switching. The hybrid control policies, used in situations where the dynamic limitations invalidate our basic control policy given in (3), are designed to apply maximum acceleration to move the system towards the domain of (3).

The lowest control policy in the partial order, which we term the *Save* control policy, is designed to prevent violations of the cell boundaries if the velocity is too high given the constraints of the system. We term the set of all states for which the Save control policy prevents a violation as the *savable set* [9]. The savable set includes the domains of the other hybrid control policies. The biggest challenge is to make sure that the system exits one cell with a velocity within the savable set of the next cell. The scaling method proposed to handle acceleration constraints will also be used to guarantee that cells connect appropriately, which will then allow us to backchain the velocity constraints for the savable set. For more information, see [1].

IV. Conclusion

The work presented in this paper represents the initial steps in a program of research designed to bring about automatic methods of deploying robust low-level control policies in constrained environments. Our goal is to enable global behaviors through composition of low-level controls in a manner that guarantees performance. This paper presents methods to accomplish this goal for fully actuated systems in $\mathbb{R}^2$; extensions to higher dimensions are given in [1].

After finalizing the constrained dynamical controls, the next step in our research plan is to extend this methodology to systems with traditional non-holonomic constraints. This will allow us to decompose a larger more complicated control problem into a series of well defined and more tractable control problems. Our long term goal is to develop methods of encoding behavior design, and facilitating automated deployment of these component control policies to enable high-level goals to be accomplished, while leveraging the robustness and performance of low-level controls to accomplish the specified tasks.

V. Acknowledgment

This work was supported by the Army Research Office under grant DAAD19-02-01-0383.

VI. References

[1] David C. Conner, Alfred A. Rizzi, and Howie Choset. Construction and automated deployment of local potential functions for global robot control and navigation. Technical Report CMU-RI-TR-03-22, Carnegie Mellon University, Robotics Institute, Pittsburgh, Pennsylvania, USA, 2003.

[2] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[3] Reid Simmons. The curvature-velocity method for local obstacle avoidance. In *IEEE International Conference on Robotics and Automation*, April 1996.

[4] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5:90–98, 1986.

[5] B. H. Krogh. A generalized potential field approach to obstacle avoidance control. In *SME Conf. Proc. Robotics Research: The Next Five Years and Beyond*, Bethlehem, Pennsylvania, August 1984.

[6] E. Rimon and D. E. Kodischek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, October 1992.

[7] C. I. Connolly. Applications of harmonic functions to robotics. In *IEEE International Symposium on Intelligent Control*, pages 498–502, August 1992.

[8] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. *International Journal of Robotics Research*, 18(6):534–555, 1999.

[9] Alfred A. Rizzi. Hybrid control as a method for robot motion programming. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 832 – 837, May 1998.

[10] Arthur Quaid and Alfred A. Rizzi. Robust and efficient motion planning for a planar robot using hybrid control. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 4021 – 4026, April 2000.

[11] Milos Zefran and Joel Burdick. Stabilization of systems with changing dynamics by means of switching. In *IEEE Conference on Robotics and Automation*, volume 2, pages 1090–1095, Leuven, Belgium, May 1998.

[12] João P. Hespanha and A. Stephen Morse. Stabilization of nonholonomic integrators via logic-based switching. *Automatica 35*, pages 385–393, 1999.

[13] J. M. Keil. Decomposing a polygon into simpler components. *SIAM J. Comput.*, 14:799–817, 1985.