To appear in the Proceedings of the 2000 IEEE
International Conference on Robotics & Automation
San Francisco, California · April 2000

Robust and Efficient Motion Planning for a Planar Robot using Hybrid Control

Arthur E. Quaid and Alfred A. Rizzi
Microdynamic Systems Laboratory*
The Robotics Institute
Carnegie Mellon University
{aquaid,arizzi}@ri.cmu.edu

Abstract

An existing hybrid control strategy is extended and applied to the control of a novel courier robot based on planar (Sawyer) motor technology augmented with integral sensing. The controller is designed to accomodate actuator limits and the desired motion characteristics, eliminating the need for careful matching of trajectory parameters with controller gains. Controller extensions allow for the unusual actuator constraints of the planar motors and independent adjustment of the parameters for each axis. Simulation results are presented for several cases to demonstrate the diversity of situations within the capability of the controller.

1 Introduction

Robot motion planning and control have traditionally been treated independently, with the controller having no knowledge of workspace boundaries and obstacles, and the planner having no knowledge of the details of control. Although this approach can often lead to simple solutions with acceptable performance, if the planning and control problems are coupled, forced separation will yield substandard results.

Recently, hybrid control techniques have been used to move higher-level decisions into the controller in a provably stable manner. Time-varying system dynamics [1], state and control constraints [2, 3], disturbances [4], and task-level programming of dynamic tasks [5] have been incorporated into controllers.

The application discussed here concerns robotic *couriers* used for carrying product sub-assemblies through a rapidly deployable assembly system called *minifactory* [6]. The minifactory is composed of a series of tabletop sized platen tiles that form the backbone of an assembly line. Overhead devices such as 2-DOF manipulator robots are attached to modular bridges spanning the platens. In addition to transporting product between overhead devices, the couriers also transiently form 4-DOF virtual machines by cooperating with these overhead devices to perform assembly operations. To eliminate the need for a central computing resource, factory programs are distributed, with each agent making decisions based on its own sensors and communication with its peers.

As multiple couriers share the platen, a distributed collision avoidance scheme [7] is necessary. The courier task considered here is the transportation of products between overhead devices. The traditional approach would be to consider this as a planning problem and choose a collision-free set of trajectories for the couriers. However, disturbances may make it impossible to track an aggressive time-parameterized trajectory, resulting in controller saturation and overshoot which could lead to collisions. These problems arise because an exact spatial and temporal path for collision-free motion is an overspecific solution since generally any efficient collision-free motion would be acceptable.

An alternative, conceptually simple approach for collision avoidance is to pre-define reservation areas in the common workspace of the agents, where an agent would only move into a new region if it can obtain exclusive access by negotiating with its peers. If a controller can then be designed that will provide efficient motion towards a goal state in a region while guaranteeing that the robot will not leave the region, explicit trajectories can be avoided, yielding feedback-based motion that will robustly respond to a large variety of disturbances. The bulk of the present work concerns the development of such a controller, restricted to a convex domain. While convex domains are not sufficient for complex reservation area layouts, multiple controllers can be used, as detailed more formally elsewhere [3], to form polygonal domains by overlapping them such that the goal of one controller lies in the domain of the next controller, eventually leading to the goal of the overall polygonal domain.

*http://www.cs.cmu.edu/~msl

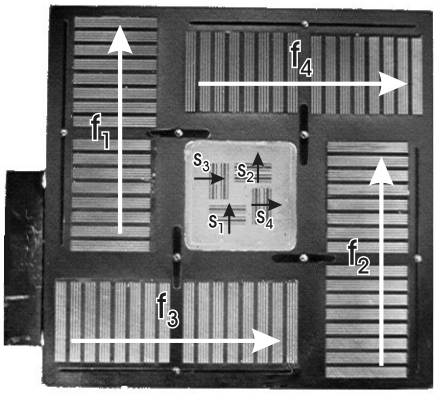

Figure 1: A planar motor with integral sensing: four linear actuators combine to generate forces and a torque in the plane.

2 System description

In the planar motor (Sawyer motor), four linear motors are combined in a single housing with air bearing channels, allowing for frictionless motion in two translational directions. It has only a small rotation range, with the actuator force capability decreasing with skew angle, reaching zero at $\pm 1.8°$. A tether is required for electrical and pneumatic connections. An alternating-current magnetic sensor [8] has been developed to enable closed-loop control [9] with programmable damping and stiffness characteristics and to improve reliability.

The courier can move at 1 m/s velocity and achieve 30 m/s^2 acceleration, with sub-micron precision over the entire platen workspace. It has one moving part, direct-drive actuators, and an air bearing, so the actuators can be driven to their limits without concern for drivetrain wear or structural vibrations.

The inequality constraints of the four actuator outputs, when mapped to the 3-D wrench space (two forces and one torque), combine to form a convex region (specifically, a rhombic dodecahedron [9]) of achievable controller outputs, making it difficult to parsimoniously represent the performance bounds.

The courier control problem can now be summarized. Assuming an unoccupied convex region on the platen, find a controller that takes the courier from any starting position within the region and a large selection of velocities to the goal position of that region without leaving the region or violating the velocity, acceleration, or actuator limits. The controller must also regulate the courier rotation to prevent reductions in the force output. The limited actuator capabilities must be carefully shared between these two tasks.

3 Controller

For simplicity, the courier dynamics are modeled as

$$\ddot{x} = u, \tag{1}$$

with $u, x \in I\!R^3$. The actuator and velocity limits are

$$\Upsilon(u) \leq 1 \tag{2}$$
$$\|\dot{x}\|_V \leq 1, \tag{3}$$

where $\|x\|_V := (x^T V x)^{1/2}$. The function $\Upsilon(u)$ is a norm-like function with the property that $u/\Upsilon(u)$ gives the largest achievable acceleration in the direction of u, computed as

$$\Upsilon(u) := \max(q_1^T u, q_2^T u, ...q_{12}^T u). \tag{4}$$

Each $q_i \in I\!R^3, i \in 1...12$ encodes the direction and distance from the origin of a face of the convex acceleration limit envelope.

A controller will also have an associated convex domain that restricts the courier position

$$x(t) \in \mathcal{P} \; \forall t \geq t_0 \tag{5}$$
$$\mathcal{P} := \{x \in I\!R^3 \mid \beta_i(x) \geq 0\} \; \forall i \in \{1...N\} \tag{6}$$
$$\beta_i := l_i^T x - c_i, \tag{7}$$

where $\|l_i\| = 1 \forall i \in \{1...N\}, l_i^T l_j \neq 1 \forall i \neq j \in \{1...N\}$, and $N \geq 4$. Note that $\mathcal{P}$ includes both translation and angle constraints.

For reasons described below, the boundary directions l_i and velocity norm $\|\cdot\|_V$ must be related by

$$(D_x \|x\|_V)^T l_i = 0 \;\; \forall x \in I\!R^3 \; | x^T l_i = 0, \tag{8}$$

where D_x is the partial derivative operator relative to x. Examples that satisfy (8) include a standard 2-norm with any boundaries, and a scaled 2-norm with its principle axes parallel to the boundaries.

The strategy for the controller design is to start with a distance function γ such as

$$\gamma(x) := \sum_{i \in 1...3} \frac{\|x_i - x_i^*\|^2}{\|x_i - x_i^*\| + \alpha_i}, \tag{9}$$

where x^* is the desired rest position for the system. Other choices for γ are possible, provided the function is positive definite about x^*, has a bounded gradient, and a positive definite Hessian, $D_x^2 \gamma$. The transposed Jacobian (gradient) of γ, $D_x \gamma^T$, can then be used as a position dependent velocity command

$$\hat{v}(x) = -V^* D\gamma^T(x), \tag{10}$$

where the symmetric, positive definite matrix V^* is used to scale $D\gamma$ such that $\|\hat{v}(x)\|_V < 1 \; \forall x \in \mathcal{P}$.

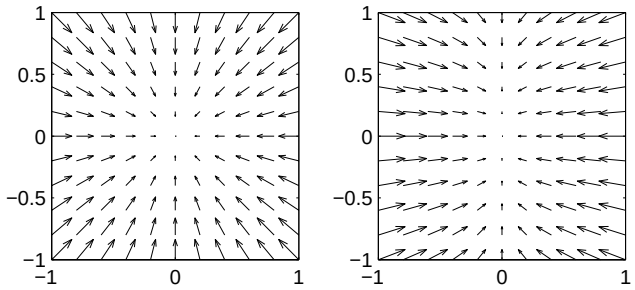

Figure 2: A vector field $\hat{v}$ derived from (9) defines a desired velocity at each position for the controller. (left: $\alpha_1 = \alpha_2 = 5$, right: $\alpha_1 = 2, \alpha_2 = 8$)

Figure 2 shows an example 2-D vector field, $\hat{v}(x)$, for two different choices of the α parameters.

Three controllers are combined such that $(x^*, 0)$ is reached without violating constraints (2)-(5). A *velocity* controller regulates the velocity to that given by $\hat{v}(x)$, but requires that the velocity already points roughly in the correct direction. This controller is sufficient if the overall controller is always activated with $\dot{x} = 0$, but it is desirable for the controller to activate at non-zero velocities for more efficient transitions between sequences of convex controllers.

Thus, an additional controller is necessary. Ideally, this controller would start from any $x \in \mathcal{P}$ and any velocity and bring the system into the domain of the velocity controller. However, there will be starting conditions that will inevitably lead to boundary violations. It is necessary to define the *savable set*, $\mathcal{S}$, as those $(x, \dot{x})$ for which a controller exists that brings $\dot{x}$ to zero without violating constraints (2)-(5). It is difficult to design a single controller that has a domain of $\mathcal{S}$, as it requires careful consideration of the boundaries, while also efficiently bringing $\dot{x}$ to $\hat{v}(x)$. However, a *stop* controller designed to bring $\dot{x}$ to zero for any starting condition in $\mathcal{S}$ can be combined with a *join* controller that seeks to take over from the stop controller as soon as possible and bring $\dot{x}$ into the domain of the velocity controller.

The domain of the stop controller is the entire savable set $\mathcal{S}$, which contains the domain of the join controller, which contains the domain of the velocity controller. The sequencing of these controllers is designed to be monotonic–the stop controller leads to the join controller leads to the velocity controller, with no "backwards" transitions. The remainder of this section presents these controllers more formally.

The use of such a control strategy was proposed previously [3], but as detailed below, the join controller has been modified to have a much larger domain, allowing it to take over from the stop controller much sooner. The implications of the more complicated actuator bounds of the couriers are also addressed.

3.1 Stopping controller (Φ_S)

Unfortunately, the form of the acceleration constraint $\Upsilon(u)$ complicates the formulation and demonstration of a controller with a domain of $\mathcal{S}$. The arguments in prior work [3] rely on having 2-norm acceleration limits. While these arguments have been extended to include scaled 2-norms, their extension to a general convex limit such as $\Upsilon(u)$ remains in progress. Note that the stopping controller's primary purpose is to extend the domain of the overall system to $\mathcal{S}$ to improve transitioning performance between convex controllers in certain cases. However, for the courier control problem, this mode of operation is not typically excited and the stop controller can be omitted.

3.2 Velocity Regulation (Φ_V)

The velocity field given by $\hat{v}(x)$ is used as a position dependent velocity command for (1) in the controller

$$u = -k\sigma(x, \dot{x})(\dot{x} - \hat{v}(x)) + \zeta(x, \dot{x}) D\hat{v}(x)\dot{x}, \qquad (11)$$

where k is a positive scalar gain for the "velocity regulation" term, which is scaled by $\sigma(x, \dot{x})$, as defined below. The second term acts as a feedforward acceleration to track the desired velocity surface $\mathcal{V} := \{(x, \dot{x}) \in T\mathcal{P} | \dot{x} - \hat{v}(x) = 0\}$, where $T\mathcal{P}$ is the tangent bundle of $\mathcal{P}$.

The domain of the controller consists of the union of two sets, the first a neighborhood of $\mathcal{V}$,

$$\mathcal{N}_v := \left\{ (x, \dot{x}) \in T\mathcal{P} \left| \frac{1}{2}(\dot{x} - \hat{v})^T (\dot{x} - \hat{v}) < \epsilon_v \right. \right\}, \qquad (12)$$

and the second the set

$$\mathcal{N}_w := \{(x, \dot{x}) \in T\mathcal{P} | 0 \leq \zeta \leq 1, \|w\| < \epsilon_w\}, \qquad (13)$$

where $\zeta := \frac{\hat{v}^T \dot{x}}{\hat{v}^T \hat{v}}$ and $w := (I - \frac{\hat{v}\hat{v}^T}{\hat{v}^T \hat{v}})\dot{x}$ are used to decompose $\dot{x}$ as $\zeta\hat{v} + w$. Note that $\mathcal{N}_w$ is a neighborhood of the states where $\dot{x}$ is "aligned with" and "smaller than" $\hat{v}$. The parameter ϵ_w is chosen such that $\Upsilon(\zeta D\hat{v}\dot{x}) < 1$ for $(x, \dot{x}) \in \mathcal{N}_w$. The scale parameter σ is set to 1 for $(x, \dot{x}) \in \mathcal{N}_v$; otherwise it is dynamically chosen to scale the feedback term of (11) by the range $0 < \sigma \leq 1$ so that $\Upsilon(u) = 1$. Parameter ϵ_v is chosen such that $\Upsilon(u) \leq 1 \ \forall (x, \dot{x}) \in \mathcal{N}_v$. It is then straightforward to show[1] that (2) is satisfied over the controller domain $\mathcal{N}_v \cup \mathcal{N}_w$.

[1] One necessary assumption is that the parameters of (9) and (10) are chosen such that $\Upsilon(u) < 1 \ \forall (x, \dot{x}) \in \mathcal{V}$.

Stability of this controller in the absence of any external constraints can be seen by considering

$$\eta_\gamma = \gamma(V^* K x) + \frac{1}{2}\dot{x}^T\dot{x} \tag{14}$$

as a candidate Lyapunov function for (1) under the influence of (11) about x^*. Taking the time derivative of (14) along the trajectory of the system yields

$$\dot{\eta}_\gamma \;\; = \;\; D\gamma V^* K\dot{x} + \dot{x}^T u, \tag{15}$$

and substituting (10) and (11) gives

$$\dot{\eta}_\gamma = -k(1-\sigma)\hat{v}^T\dot{x} - k\sigma\dot{x}^T\dot{x} + \zeta\dot{x}^T D\hat{v}\dot{x}. \tag{16}$$

The first term is identically zero for $(x,\dot{x}) \in \mathcal{N}_v$ (because $\sigma = 1$) and negative semidefinite in the state for $(x,\dot{x}) \in \mathcal{N}_w$ (because $\hat{v}^T\dot{x} \geq 0$). The remaining terms are also negative semidefinite in the state, and we can conclude that $\lim_{t\to\infty}\dot{x} = 0$. Finally, by recourse to LaSalle's Invariance Principle [10], $\lim_{t\to\infty}\gamma(V^*Kx) = 0$ and x converges to x^*.

Proposition 1 *Under the influence of (11) the surface* $\mathcal{W}$:=$\{(x,\dot{x}) \in T\mathcal{P} | w = 0\}$ *is both attractive and invariant over* $\mathcal{N}_v \cup \mathcal{N}_w$.

This can be easily seen by considering the Lyapunov function

$$\eta_w = \frac{1}{2}w^T w. \tag{17}$$

After taking the derivative of (17) along the trajectories of the system under the control (11), it can be shown that

$$\dot{\eta}_w \;\; = \;\; -k\sigma w^T w, \tag{18}$$

which allows us to conclude that w converges to zero.□

Proposition 2 *Under the influence of (11) the surface* $\mathcal{D}$:=$\{(x,\dot{x}) \in T\mathcal{P} | \zeta = 1\}$ *is both attractive and invariant over* $\mathcal{W}$.

Similarly, this can be seen by considering the Lyapunov function

$$\eta_\zeta = \frac{1}{2}(\zeta - 1)^2. \tag{19}$$

After taking the derivative of (19) along the trajectories of the system under the control (11), it can be shown that

$$\dot{\eta}_\zeta \;\; = \;\; (\zeta - 1)\frac{\dot{x}^T D\hat{v}}{\hat{v}^T\hat{v}} w - (\zeta - 1)^2 k\sigma. \tag{20}$$

The first term is zero for $(x,\dot{x}) \in \mathcal{W}$, and the second is negative definite in $(\zeta - 1)$, so we can conclude that ζ converges to 1.□

Combining these propositions, it is easy to show that the controller (11) will drive the state to $\mathcal{W}$, and then to $\mathcal{W}\cap\mathcal{D} = \mathcal{V}$, and we can conclude that $\mathcal{V}$ is both attractive and invariant for the entire controller domain, $\mathcal{N}_v \cup \mathcal{N}_w$.

The design of this controller requires choosing the velocity limit V^*, and any parameters of the particular γ form such that constraints (2)-(5) are not violated. The acceleration constraint (2) was discussed earlier. The velocity constraint (3) will be obeyed if the time derivative of $\|\dot{x}\|_V$ decreases when $\|\dot{x}\|_V = 1$. Details are omitted here, but it can be shown that (3) will not be violated under this control policy.

The final constraint on position (5) must also be examined. As the velocity controller has no knowledge of the boundaries, the designer must ensure *a priori* that they are obeyed. One problem occurs if the "trajectory" implied by $\hat{v}$ starting at any $(x,\dot{x}) \in \mathcal{V}$ leaves $\mathcal{P}$. This case can be avoided if the parameters are chosen such that

$$\hat{v}(x)^T l_i > 0 \;\; \forall i \in \{1...12\}, \{x | \beta_i(x) = 0\}, \tag{21}$$

i.e. $\hat{v}$ always points towards the interior of $\mathcal{P}$ along its boundaries. In addition, system trajectories within the domain must also not leave $\mathcal{P}$. A Lyapunov-based derivation of a suitable condition for parameters was sketched in [3], although formal demonstration of this constraint is still under development. In practice, reasonable values for V^* and α do not exhibit this defect.

3.3 Join controller (Φ_J)

The join controller has three components

$$u = \sigma_a u_a + \sigma_b u_b + \sigma_s u_s. \tag{22}$$

As defined below, u_a is designed to avoid boundaries while using a time-varying fraction σ_a of the total acceleration capability, u_b is designed to push $\zeta = \frac{\hat{v}^T\dot{x}}{\hat{v}^T\hat{v}}$ into the range $0 \leq \zeta \leq 1$, and u_s is designed to steer the velocity by reducing any undesirable components. The σ values are chosen somewhat conservatively so that (2) is not violated, with the *avoidance* taking priority over *braking*, which has priority over *steering*. Each of the three u's will be designed with the properties

$$u^T\dot{x} \;\; \leq \;\; 0 \tag{23}$$

$$u^T V\dot{x} \;\; \leq \;\; 0. \tag{24}$$

With these properties it is easy to show that the join controller will (in the worst case) bring $\dot{x}$ to zero, which is in the domain of the velocity controller. With property (24) it is easy to show that the velocity constraint (3) will not be violated.

Boundaries are avoided by computing the ratio δ_i of the required stopping distance to each boundary to the distance to that boundary

$$\delta_i = \frac{\max(-l_i^T \dot{x}, 0)^2 / 2u_{l,i}}{\beta_i}, \tag{25}$$

where $u_{l,i} = \frac{l_i}{\Upsilon(l_i)}$ is the maximum acceleration that can be generated normal to the ith boundary. The net direction of the avoidance control is given by $u_a = \sum_i \delta_i u_{l,i}$, with magnitude $\sigma_a = \sum_i \delta_i$. It is easy to show that (23) is satisfied, but u_a may act to increase the scaled velocity norm $\|\dot{x}\|_V$ in certain cases, possibly leading to a violation of (3). However, (8) ensures that the velocity norm and the boundaries are "aligned" such that this violation is impossible.

The braking control is simply defined with direction

$$u_b = \begin{cases} 0 & 0 \le \zeta \le 1 \\ -\frac{\dot{x}}{\Upsilon(\dot{x})} & \text{otherwise,} \end{cases} \tag{26}$$

and magnitude

$$\sigma_b = \begin{cases} 1 - \sigma_a & u_b \neq 0 \\ 0 & u_b = 0. \end{cases} \tag{27}$$

The final steering control uses the remaining acceleration capability, if any, to reduce velocities normal to the local desired velocity. This normal may be defined based on a Euclidean norm or a scaled norm that reflects the velocity limits

$$u_{s1} = -\left(\mathrm{I} - \frac{\hat{v}\hat{v}^T}{\|\hat{v}\|^2}\right)\dot{x} \tag{28}$$

$$u_{s2} = -\left(\mathrm{I} - \frac{\hat{v}\hat{v}^T V}{\|\hat{v}\|_V^2}\right)\dot{x}. \tag{29}$$

Note that u_{s1} will never increase $\|\dot{x}\|$ while u_{s2} will never increase $\|\dot{x}\|_V$. Also, (we claim) at any given instant, either u_{s1} will decrease $\|\dot{x}\|_V$ or u_{s2} will decrease $\|\dot{x}\|$. In this case, the steering direction can be chosen as

$$u_s = \begin{cases} u_{s1} & \dot{x}^T V u_{s1} < 0 \\ u_{s2} & \dot{x}^T V u_{s1} \ge 0 \end{cases} \tag{30}$$

to guarantee that the steering control will not violate the velocity constraint and not increase the Euclidean velocity norm. The steering magnitude is chosen to use any available acceleration (reduced near $\Upsilon(u_s) = 0$ to prevent chattering) as

$$\sigma_s = \begin{cases} \frac{1}{\Upsilon(u_s)}(1 - \sigma_a - \sigma_b) & \Upsilon(u_s) > \epsilon_s \\ \frac{\Upsilon(u_s)}{\epsilon_s^2}(1 - \sigma_a - \sigma_b) & \Upsilon(u_s) \le \epsilon_s. \end{cases} \tag{31}$$

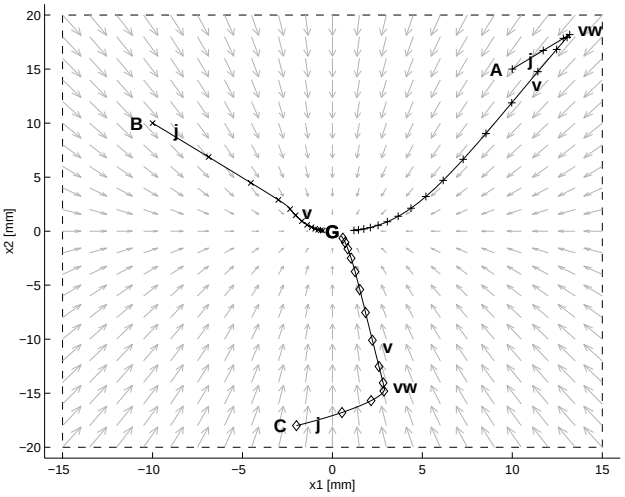

Figure 3: Position traces for three simulations A, B, C with different starting conditions but identical boundary $\mathcal{P}$ (dashed lines). Controller switches are indicated by codes (j=join, vw=velocity controller in $\mathcal{N}_w$, v=velocity controller in $\mathcal{N}_v$). Symbols (+, $\times$, $\diamond$) demarcate 5 msec time intervals.

The domain of the overall join controller is given by constraints (2), (3), and (5) together with one additional constraint, $\sum_i \delta_i < 1$, to guarantee that the avoidance controller will not activate if it might be overwhelmed. The goal of the controller, expressed in terms of ζ and w from the previous section, is the set $\{(x, \dot{x}) \in T\mathcal{P} | w = 0, (0 \le \zeta \le 1)\}$, which is a subset of the domain of Φ_V.

4 Results

Simulations with varying starting conditions were performed to demonstrate reasonable controller behavior, at least anecdotally. All simulations use the courier and controller parameters given in Table 1.

The first simulation shows a case where the controller becomes active with the courier rapidly approaching the boundary of $\mathcal{P}$. The translational position domain is shown in Figure 3 along with the vector field $\hat{v}$ and the position trace (marked with +). The position trace begins (A) with the join controller (j) being activated. In this case, the robot is headed in the wrong direction, so the join controller brings it to a stop. Next, the velocity controller with the system state in $\mathcal{N}_w$ (vw) increases the velocity, bringing the system into $\mathcal{N}_v$ (v), which leads the robot to the goal (G) at the origin. Note that controller switches occur in the proper sequence. Although not shown here, the angle also starts with an initial error and a velocity of

	x	y	θ
$L_{1,2}$	± 0.027 s^2/m	0.0 s^2/m	0.0 s^2/rad
$L_{3,4}$	0.0 s^2/m	± 0.027 s^2/m	0.0 s^2/rad
L_{5-12}	± 0.014 s^2/m	± 0.014 s^2/m	± 8.93e-4 s^2/rad
k	800 1/s	(same)	(same)
α	0.05 m	0.02 m	0.01 rad
V^*	1.0 m/s	1.0 m/s	0.5 rad/s
diag(V)	1.0 s^2/m^2	1.0 s^2/m^2	4.0 s^2/rad^2

Table 1: Simulation values for convex acceleration limits and controller parameters

the incorrect sign. However, the controller smoothly handles both the angle and translational errors.

The second simulation shows a case where the controller becomes active with the robot headed towards the goal with a velocity larger than $\hat{v}$. The position sequence (B) in Figure 3 shows the join controller starting with its braking component active to reduce ζ. After sufficient braking, the system state directly enters $\mathcal{N}_v$ and the velocity controller (v) takes over and moves the robot to the origin.

The final simulation shows the robot headed roughly perpendicular to the goal with a large velocity. Position sequence (C) in Figure 3 shows the join starting with its braking component active for a short time followed by the steering component. In this case, a component of the initial velocity pointed towards the goal, so that complete stopping was unnecessary. In practice, we expect this case to be the most common when multiple convex region controllers activate one another in succession.

5 Conclusions

Control techniques introduced in [3] have been extended for the courier control problem. Convex acceleration limits and individual tuning of multiple axes are now possible. A join controller with a much larger domain was presented, eliminating the need for the stopping controller in many situations.

Simulation results demonstrated good performance in a wide range of situations, simultaneously avoiding region boundaries, angle limits, and actuator limits.

We have begun controller experiments using a minifactory courier. An algorithm for computing the appropriate velocity controller domain parameter ϵ_v was derived. It was fairly easy to find appropriate controller parameters for the translational axes, but the small motion range of the rotational axis required more care. The controller performed well overall, but occasionally the monotonicity of the controller sequences was violated, presumably due to sensing inaccuracies, resulting in some minor momentary chattering.

Acknowledgments

The authors acknowledge the contributions of Ralph Hollis, Zack Butler, and other Microdynamic Systems Laboratory members. This work was supported in part by the NSF under grants CDA-9503992, DMI-9527190, and DMI-9523156. Quaid was supported in part by a Lucent Technologies Fellowship.

References

[1] M. Zefran and J. W. Burdick, "Design of switching controllers for systems with changing dynamics," in *Proceedings of the 37th IEEE Conference on Decision and Control*, (Tampa, FL), pp. 2113–2118, Dec. 1998.

[2] I. Kolmanovsky, E. G. Gilbert, N. H. McClamroch, and T. L. Maizenberg, "Toward less conservative designs of multimode controllers for systems with state and control constraints," in *Proceedings of the 37th IEEE Conference on Decision and Control*, (Tampa, FL), pp. 1647–1652, Dec. 1998.

[3] A. A. Rizzi, "Hybrid control as a method for robot motion programming," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, pp. 832–837, May 1998.

[4] I. Kolmanovsky and E. G. Gilbert, "Multimode regulators for systems with state and control constraints and disturbance inputs," in *Control Using Logic-Based Switching*, (Block Island, RI), pp. 104–117, 1995.

[5] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *International Journal of Robotics Research*, vol. 18, pp. 534–555, June 1999.

[6] R. L. Hollis and A. Quaid, "An architecture for agile assembly," in *American Society of Precision Engineering 10th Annual Meeting, Austin, Texas*, pp. 372–375, October 1995.

[7] A. A. Rizzi, J. Gowdy, and R. L. Hollis, "Agile assembly architecture: an agent based approach to modular precision assembly systems," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, pp. 1511–1516, April 1997.

[8] Z. J. Butler, A. A. Rizzi, and R. L. Hollis, "Integrated precision 3-DOF position sensor for planar linear motors," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, pp. 3109–3114, May 1998.

[9] A. E. Quaid and R. L. Hollis, "3-DOF closed-loop control for planar linear motors," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, pp. 2488–2493, May 1998.

[10] J. P. LaSalle, "Some extensions of Liapunov's second method," *IRE Transactions on Circuit Theory*, pp. 520–527, December 1960.