Stabilization of Trajectories for Systems with Nonholonomic Constraints

G. Walsh, D. Tilbury, S. Sastry, R. Murray *, J.P. Laumond †
Electronics Research Laboratory
Department of Electrical Engineering and Computer Sciences
University of California Berkeley, CA 94720

Abstract

A new technique for stabilizing nonholonomic systems to trajectories is presented. It is well known (see [2]) that such systems cannot be stabilized to a point using smooth-static state feedback. In this paper we suggest the use of control laws for stabilizing a system about a trajectory, instead of a point. Given a nonlinear system and a desired (nominal) feasible trajectory, the paper gives an explicit control law which will locally exponentially stabilize the system to the desired trajectory. The theory is applied to several examples, including a car-like robot.

1 Introduction

There has been a great deal of recent research on the problem of stabilizing a system with nonholonomic (nonintegrable) constraints on its velocities [1, 5, 12]. Of course, by Brockett's necessary conditions for stability, one may demonstrate that systems with nonintegrable velocity constraints cannot be stabilized to a point with smooth static-state feedback [3]. Given this result, researchers have offered both non-smooth feedback laws [6] and time-varying feedback laws [12] for stabilizing simple mobile robots to a point. However, it is fair to say that these approaches are not yet fully general.

Our approach is to stabilize about *trajectories* instead of points. Given a feasible trajectory for the system generated by an open-loop path planner, we can compute the linearization of the system about this nominal trajectory. If the linear time-varying system thus obtained is uniformly completely controllable in a certain sense (to be made explicit in §2), we define a linear time-varying

*Address: Department of Mechanical Engineering, California Institute of Technology, Pasadena, CA 91125

†Address: Laboratoire d'Automatique et d'Analyse des Systèmes, Toulouse, France

feedback law which will locally stabilize the system about this nominal trajectory.

Thus, the problem this paper solves is: given a nonholonomic system, a feasible desired trajectory to follow, a known clearance between obstacles, and a measure of accuracy of the sensors, find a control law which will stabilize the system to this path, avoiding the obstacles robustly in the face of disturbances.

In §2 we present our control law and show it to be exponentially convergent. In the following sections, we apply this control law to various nonholonomic systems, including the system generated by the Heisenberg control algebra, a wheeled mobile robot called Hilare, and a front wheel drive car.

In the examples, we focus on mobile robots with an objective of creating a composite controller that will: first, have off-line computation of a trajectory which avoids the obstacles [9]; second, apply the control law given here to stabilize the system to the open loop collision-free trajectory; third, while executing, use sensors to detect possible collisions due to poor *a priori* information. In this case, new information can be used to update the model of the environment and restart the process. Such a controller would be able to reject many types of disturbances including noise in the sensors, initial condition errors, and errors introduced along the trajectory.

2 An Exponentially Stabilizing Control Law

We consider a system:

$$\begin{aligned} \dot{x} &= f(x,u) \qquad\qquad (1) \\ x &\in \mathbb{R}^n \quad \text{states} \\ u &\in \mathbb{R}^p \quad \text{inputs} \\ x^0(t) & \qquad \text{Desired trajectory} \\ u^0(t) & \qquad \text{Nominal Inputs} \end{aligned}$$

with $f(x,u)$ C^2 with respect to x and u.

Remarks: We shall focus on systems where $f(x,0)$ is identically zero. Systems like this are called "drift-free" and encompass most of the models used in the literature. However, the method we present here is general enough to include systems like (1) which have non-zero drift terms. Thus the proofs will include the drift terms, although the worked examples are all drift-free.

Inspired by the result on linear systems found in [4], we have picked the following control law:

Proposition 1 (A Stabilizing Control Law)
Given a system of the form (1), a desired trajectory $x^0(\cdot)$, and a nominal input $u^0(\cdot)$, both bounded for all t, define the following:

$$A(t) := \frac{\partial f}{\partial x}(x^0(t), u^0(t))$$

$$B(t) \quad := \quad \frac{\partial f}{\partial u}(x^0(t), u^0(t))$$

Define $\Phi(t, t_0)$, contained at each time t in $\mathbb{R}^{n\times n}$ to be the solution to the differential equation $\dot{\Phi}(t, t_0) = A(t)\Phi(t, t_0)$ with $\Phi(t_0, t_0) = I$. Further, define for some $\alpha > 0$:

$$H_c(t_0, t) \quad = \quad \int_{t_0}^{t} e^{6\alpha(t_0-\tau)}\Phi(t_0, \tau)B(\tau)B(\tau)^T\Phi(t_0, \tau)^T d\tau$$

If there exists a δ such that $H_c(t, t+\delta)$ is bounded away from singularity [1] *for all t, then define $P_c(t)$ as follows:*

$$P_c(t) \quad := \quad H_c^{-1}(t, t+\delta)$$

Now, if there exist two numbers p_c^m, p_c^M such that:

$$0 \quad < \quad p_c^m I < P_c(t) < p_c^M I \qquad \forall t \in \mathbb{R}_+$$

Then, for any function $\gamma(t) : \mathbb{R}_+ \to [\frac{1}{2}, \infty)$, continuous and bounded, the linear time varying feedback law:

$$u \quad = \quad u^0 - \gamma(t)B(t)^T P_c(t)\left(x - x^0\right)$$

locally, uniformly, exponentially stabilizes the system (1) to the desired trajectory $x^0(t)$ at a rate greater than $2\alpha p_c^m(p_c^M)^{-1} > 0$.

Proof: First, define the error signal e and error input v as:

$$\begin{aligned} e &= x - x^0 \in \mathbb{R}^n \\ v &= u - u^0 \in \mathbb{R}^p \end{aligned}$$

We solve for the dynamics of these error signals using the Taylor Series expansions:

$$\begin{aligned} \dot{e} &= f(x^0 + e, u^0 + v) - f(x^0, u^0) \\ &= A(t)e + B(t)v + \text{h.o.t.} \end{aligned}$$

All terms with dependencies on x^0, u^0 will be rewritten as functions of time. In addition to $A(t), B(t)$, define $o(e, v, t)$ to be the higher order terms

$$o(e, v, t) \quad = \quad f(x^0 + e, u^0 + v) - f(x^0, u^0) - A(t)e - B(t)v$$

Note that since $v = -\gamma(t)B^T(t)P_c(t)e$, we may rewrite $o(e, v, t)$ so it depends only on e, t; call this $\hat{o}(e, t)$. As $x^0(t), u^0(t)$ are bounded for all t, $B(t)$ is bounded

[1] If the linear time-varying system is uniformly completely controllable over intervals of length $\delta > 0$ then $H_c(t, t+\delta)$ is uniformly invertible.

for all t which implies that $||\gamma(t)B^T(t)P_c(t)e|| \leq K||e||$ for some $K < \infty$ as $\gamma(t), P_c(t)$ are also bounded for all t. Given this we may show that:

$$\lim_{||e|| \to 0} \sup_{t \geq 0} \frac{||\hat{o}(e,t)||}{||e||} = 0 \tag{2}$$

Thus:

$$\begin{aligned} \dot{e} &= A(t)e + B(t)v + o(e,v,t) \\ \dot{e} &= \tilde{\mathrm{A}}(t)e + \hat{o}(e,t) \end{aligned} \tag{3}$$

with:

$$\tilde{\mathrm{A}}(t) = A(t) - \gamma(t)B(t)B(t)^T P_c(t)$$

Inspired by [4], we pick a Lyapunov function:

$$V(e,t) = e^T P_c(t) e \tag{4}$$

and calculate its time derivative along trajectories of the system (3). One may verify that:

$$\begin{aligned} \dot{P}_c(t) = & -6\alpha P_c(t) - P_c(t)A(t) - A^T(t)P_c(t) \\ & + P_c(t)[B(t)B^T(t) - e^{-4\alpha\delta}\Phi(t,t+\delta)B(t+\delta)B^T(t+\delta)\Phi^T(t,t+\delta)]P_c(t) \end{aligned}$$

Thus the time derivative of the Lyapunov function is:

$$\begin{aligned} \dot{V}(e,t) = & -e^T[6\alpha P_c(t) + (2\gamma(t) - 1)P_c(t)B(t)B^T(t)P_c(t)]e \\ & -e^T e^{-4\alpha\delta} P_c(t)\Phi(t,t+\delta)B(t+\delta)B^T(t+\delta)\Phi^T(t,t+\delta)P_c(t)e \\ & +2e^T P_c(t)\hat{o}(e,t) \end{aligned} \tag{5}$$

Note that if $\gamma(t) \geq \frac{1}{2}, \forall t$, then the first two terms in (5) are less than or equal to $-6\alpha p_c^m ||e||^2$. Secondly, because of (2), there exists a number $\epsilon > 0$ such that:

$$||\hat{o}(e,t)|| \leq \alpha p_c^m (p_c^M)^{-1} ||e||, \forall e \text{ such that } ||e|| \leq \epsilon$$

This implies that:

$$|2e^T P_c(t)\hat{o}(e,t)| \leq 2\alpha p_c^m ||e||^2, \forall ||e|| \leq \epsilon$$

Thus, now similar to [4], we may say that:

$$\dot{V}(e,t) \leq -4\alpha p_c^m ||e||^2 \tag{6}$$

Further we may state that:

$$\dot{V}(e,t) \leq -4\alpha p_c^m (p_c^M)^{-1} V(e,t) \tag{7}$$

Finally we may conclude:

$$\begin{aligned} V(e,t) &\leq V(e_0,t_0)e^{-4\alpha p_c^m (p_c^M)^{-1}(t-t_0)} \\ \|e(t)\| &\leq \|e(t_0)\|\sqrt{\frac{p_c^M}{p_c^m}}e^{-2\alpha p_c^m (p_c^M)^{-1}(t-t_0)} \end{aligned} \tag{8}$$

which gives us the specified convergence rate for the error signals $\square$.

Remark 1: This convergence rate may be shown to be independent of p_c^m and p_c^M. To demonstrate this, define z, y:

$$\begin{aligned} z &= ey \\ y &\in \mathbb{R} \qquad \text{with} \\ \dot{y} &= \alpha y \end{aligned}$$

Now we wish to solve for the dynamics of z.

$$\dot{z} = (\tilde{\mathrm{A}}(t) + \alpha I)z + \hat{o}(e,t)y$$

We will pick the same Lyapunov equation and calculate its derivative, using the same arguments as before:

$$\begin{aligned} V(z,t) &= z^T P_c(t) z \\ \dot{V}(z,t) &\leq -4\alpha p_c^m \|z\|^2 + 2z^t P_c(t)\hat{o}(e,t)y \end{aligned}$$

Given the exponential convergence of e when it starts sufficiently close to the origin, we may say that after some time T the last factor may be bounded as follows:

$$|2zP_c(t)\hat{o}(e,t)y| \leq 2\alpha p_c^m \|z\|^2$$

Thus we may write, as before, that:

$$\dot{V}(z,t) \leq -2\alpha p_c^m \|z\|^2$$

And following equations (8) and (9) we will obtain the same convergence rate, $\alpha p_c^m (p_c^M)^{-1}$. However, we may note that:

$$\|z\| = e^{\alpha(t-t_0)}|y_0|\|e\|$$

Thus, if z is exponentially convergent at a rate $\alpha p_c^m (p_c^M)^{-1}$ after some time T, then e is exponentially convergent at a rate $\alpha p_c^m (p_c^M)^{-1} + \alpha > \alpha$ after some time T, thus for a sufficiently large k we may state that:

$$\|z\| \leq ke^{-\alpha(t-t_0)}\|z_0\|\square$$

Remark 2: For some regulator applications, it is desirable not to need information on the future trajectory of the system. To deal with this concern, define $P_r(t)$, similar to $P_c(t)$, again assuming the inverse in the formula exists:

$$P_r(t) \quad = \quad (H_c(t, t-\delta))^{-1}$$

Notice that this matrix is dependent on past values of the trajectory and not on future values. As before, if there exists two numbers p_r^m and p_r^M such that :

$$0 \quad < \quad p_r^m I < P_r(t) < p_r^M I \qquad \forall t \in \mathbb{R}_+$$

then for any $\gamma(t) : \mathbb{R}_+ \to [\frac{1}{2}, \infty)$, continuous and bounded, the linear time varying feedback law:

$$u \quad = \quad u^0 - \gamma(t) B(t)^T P_r(t)(x - x^0)$$

locally uniformly exponentially stabilizes the system (1) at a rate greater then α.

The proof is similar to the last control law and so will be left to the interested reader. It is useful to note that:

$$\begin{aligned} \dot{P}_r(t) \quad = \quad & -6\alpha P_r(t) - P_r(t)A(t) - A^T(t)P_r(t) \\ & + P_r(t)[B(t)B^T(t) - e^{4\alpha\delta}\Phi(t, t-\delta)B(t-\delta)B^T(t-\delta)\Phi^T(t, t-\delta)]P_r(t) \end{aligned}$$

We have applied this control law to three example systems. The first is chosen because its simple structure allows for the explicit computation of the control laws. The second is the Hilare-like mobile robot, without drift, and the third example is a front wheel drive car.

3 Example: the Heisenberg Control Algebra

Here we will consider one of the simplest nonholonomic systems: the system whose control Lie algebra is the Heisenberg algebra with two generators [2]. The differential equations are as follows:

$$\begin{aligned} \dot{x_1} &= u_1 \\ \dot{x_2} &= u_2 \\ \dot{x_3} &= x_2 u_1 \end{aligned} \tag{9}$$

This system's straightforward structure allows us to compute the control laws in closed form. We will investigate two trajectories for this system, a "trivial" trajectory which is just a point, and a straight line.

Much of the control law can be found without reference to the specific form of the desired trajectory. The first step is to find the matrices $A(t), B(t)$.

$$
\begin{aligned}
A(t) &= \frac{\partial f}{\partial x}(x^0, u^0) \\
&= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & u_1^0 & 0 \end{bmatrix} \\
B(t) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ x_2^0(t) & 0 \end{bmatrix}
\end{aligned} \tag{10}
$$

With these, the state transition matrix associated with this particular $A(t)$ can be found. Using the fact that $\Phi(t_0, t_0) = I$ and that $\Phi(t, t_0)$ satisfies the differential equation $\dot{\Phi}(t, t_0) = A(t)\Phi(t, t_0)$, it can be shown that:

$$
\Phi(t, t_0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & x_1^0(t) - x_1^0(t_0) & 1 \end{bmatrix}
$$

Now that we have the state transition matrix, we solve for the derivative of $H_c(t_0, t)$ as follows,

$$
\begin{aligned}
\dot{H}_c(t_0, t) &= e^{4\alpha(t_0 - t)}\Phi(t_0, t)B(t)B(t)^T\Phi^T(t_0, t) \\
&= e^{4\alpha(t_0 - t)} \begin{bmatrix} 1 & 0 & x_2^0(t) \\ 0 & 1 & x_1^0(t_0) - x_1^0(t) \\ x_2^0(t) & x_1^0(t_0) - x_1^0(t) & \left(x_2^0(t)\right)^2 + \left(x_1^0(t_0) - x_1^0(t)\right)^2 \end{bmatrix}
\end{aligned}
$$

The first nominal trajectory that we choose is the trivial one, where the system stays fixed at a given point for all time. Note that stabilizing to this trajectory is equivalent to finding a point stabilization feedback law. The trajectory is highly degenerate in the sense that both nominal inputs are zero. We choose our desired point $x^0(t)$ to be the origin, $(0, 0, 0)$.

It may be shown that in this case, $H_c(t_0, t)$ has the following form (for $\alpha = 1$):

$$
H_c(t_0, t) = \begin{bmatrix} 1 - e^{t_0 - t} & 0 & 0 \\ 0 & 1 - e^{t_0 - t} & 0 \\ 0 & 0 & 0 \end{bmatrix}
$$

Thus the matrix $H_c(t, t + \delta)$ is not invertible for any choice of δ, we cannot find the matrix $P_c(t) = H_c^{-1}(t, t + \delta)$ which is used in the definition of the control law, and therefore the method presented in this paper cannot be used to stabilize the system (9) to a point.

The second sample trajectory will be less trivial. We have chosen the straight line in state space described by $x^0(t) = (0, t, 0)$, with nominal input $u^0(t) = (0, 1)$. This trajectory is somewhat degenerate in the sense that one of the nominal inputs is zero. However, since the matrix H_c is invertible, our strategy will work for this trajectory. In fact, the determinant of H_c is independent of time, as can be seen by the following formula (where $\alpha = 1$):

$$\det\left(H_c(t, t+\delta)\right) \quad = \quad 1 - e^{3\delta} + (3 + \delta^2)(e^{-2\delta} - e^{-\delta})$$

We can choose any value of δ for which the previous expression is non-zero. In our simulation, $\delta = 1$ was chosen. (Note: nearly identical simulation results are obtained for the trajectory given by $u^0(t) = (1, 0)$).

The initial error for this simulation was $(0.2, -0.3, 0.2)$. The simulation was run for 8 seconds, and the results are given in Figures (1-3). As the graphs of the state variables were in general difficult to interpret, we have instead shown the error coordinates $e(t)$ and the error inputs $v(t)$ versus time.

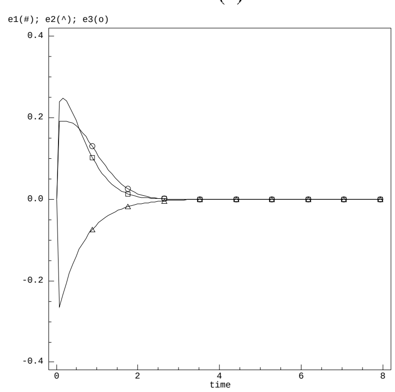

Figure 1: Plot of errors e versus time. The errors all quickly converge to zero for this path.

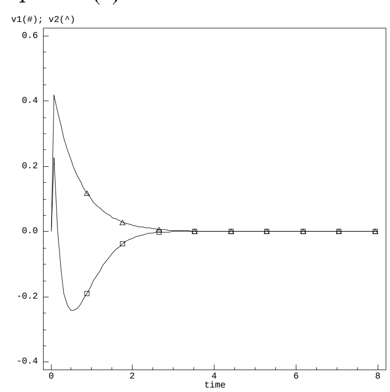

Figure 2: Graph of the error inputs v versus time. Note how all inputs are bounded and smooth.

4 Trajectory Stabilization for a Simple Nonholonomic Mobile Robot: Hilare

Hilare is a wheeled mobile robot created at LAAS, Laboratoire d'Automatique et d'Analyse des Systèmes, located in Toulouse, France [8]. This robot has two parallel wheels which can be controlled independently. By commanding the same velocity to both wheels, the robot moves in a straight line. By commanding velocities with the same magnitude but opposite directions, the robot pivots about its axis. Although the actual input is the acceleration, we are doing only a kinematic analysis and assuming that we can control the velocity. See Figure 4 for a diagram of Hilare.

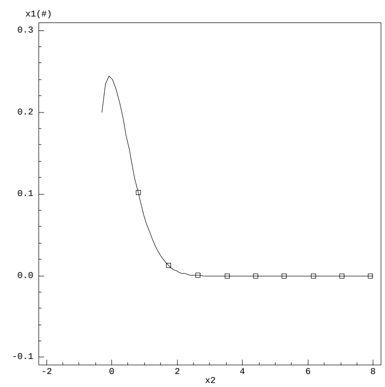

Figure 3: This x_1, x_2 phase plot shows the actual trajectory, projected onto the (x_1, x_2) plane. The desired trajectory is a straight line along the x_2 axis.

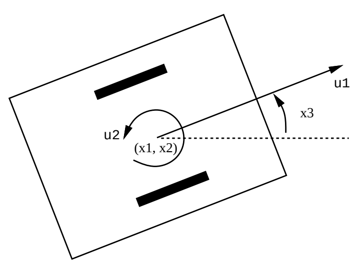

Figure 4: Model of the mobile robot Hilare.

Assuming that wheel velocities are the inputs, one may model Hilare as follows:

$$
\begin{aligned}
\dot{x}_1 &= \cos(x_3)u_1 \\
\dot{x}_2 &= \sin(x_3)u_1 \\
\dot{x}_3 &= u_2
\end{aligned} \tag{11}
$$

Note from Figure 4 that the coordinates (x_1, x_2) represent the position of the robot in the plane, and x_3 is its orientation.

Again, one would hope the system's straightforward structure allows the control laws to be computed in closed form. The first step is to find the matrices $A(t)$, contained in $\mathbb{R}^{3\times 3}$, and $B(t)$ which is contained in $\mathbb{R}^{3\times 2}$:

$$
\begin{aligned}
A(t) &= \frac{\partial f}{\partial x}(x^0, u^0) \\
&= \begin{bmatrix} 0 & 0 & -\sin(x_3^0)u_1^0 \\ 0 & 0 & \cos(x_3^0)u_1^0 \\ 0 & 0 & 0 \end{bmatrix} \\
B(t) &= \begin{bmatrix} \cos(x_3^0) & 0 \\ \sin(x_3^0) & 0 \\ 0 & 1 \end{bmatrix}
\end{aligned}
$$

Now the state transition matrix associated with this particular $A(t)$ may be found. Using the fact that $\Phi(t_0, t_0) = I$ and that $\Phi(t, t_0)$ satisfies the differential equation $\dot{\Phi}(t, t_0) = A(t)\Phi(t, t_0)$, it may be shown that:

$$
\Phi(t, t_0) = \begin{bmatrix} 1 & 0 & f_s(t, t_0) \\ 0 & 1 & f_c(t, t_0) \\ 0 & 0 & 1 \end{bmatrix}
$$

where:

$$
\begin{aligned}
f_s(t, t_0) &= \int_{t_0}^{t} -\sin\left(x_3^0(\tau)\right) u_1^0(\tau) d\tau \\
f_c(t, t_0) &= \int_{t_0}^{t} \cos\left(x_3^0(\tau)\right) u_1^0(\tau) d\tau
\end{aligned}
$$

Now that we have the state transition matrix, we can solve for the derivative of $H_c(t_0, t)$ as follows,

$$
\begin{aligned}
\dot{H}_c(t, t_0) &= e^{4\alpha(t_0 - t)}\Phi(t_0, t)B(t)B(t)^T\Phi(t_0, t) \\
&= e^{4\alpha(t_0 - t)} \begin{bmatrix} \cos^2(x_3^0) + f_s^2 & \cos(x_3^0)\sin(x_3^0) + f_s f_c & f_s \\ \cos(x_3^0)\sin(x_3^0) + f_s f_c & \sin^2(x_3^0) + f_c^2 & f_c \\ f_s & f_s & 1 \end{bmatrix}
\end{aligned}
$$

However, for the nominal trajectories x^0 that we have chosen to simulate, the integrals f_s, f_c do not have a closed form. Thus we cannot directly compute the control law, and so we must compute the matrix P_c and the control law numerically. In doing so the following identity is useful:

$$\dot{\Phi}(t, t+\delta) \quad = \quad A(t)\Phi(t, t+\delta) - \Phi(t, t+\delta)A^T(t+\delta)$$

The first nominal trajectory for this system (Hilare) is generated by the inputs $u_1 = \sin(t), u_2 = \cos(t)$. We set $\alpha = 0.1, \delta = 1.0$. After one cycle this input steers the system in the direction given by the Lie bracket of the two input vector fields, or $[g_1, g_2]$. The initial condition was chosen to be (-0.1, 0.2, 0.1), and the simulation was run for 2π seconds. See Figures (5 - 7) for results.

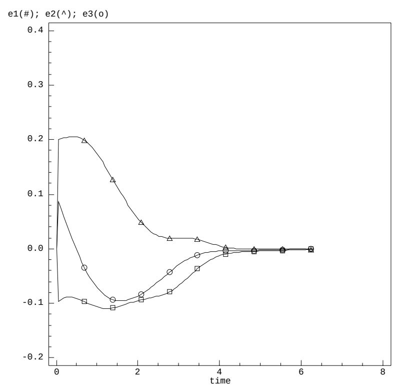

Figure 5: Errors e versus time.

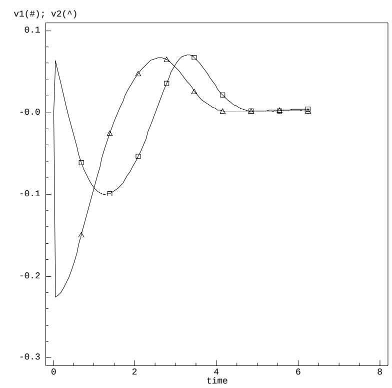

Figure 6: Error inputs v versus time.

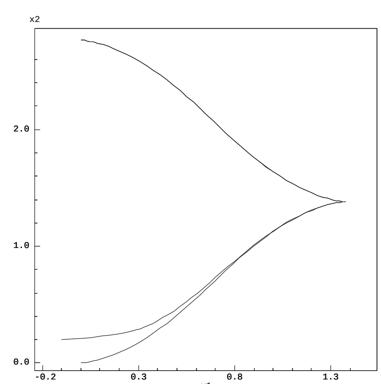

Figure 7: This phase plot shows the nominal and actual trajectories projected onto the (x_1, x_2) plane (the orientation of the robot is not shown). The desired trajectory starts at (0,0) whereas the actual trajectory has an initial offset of (-0.1,0.2). Note how quickly and smoothly the system converges to the desired trajectory.

The second nominal trajectory to which we have applied our stabilization procedure is a circular path. This choice was inspired by the work of Reeds and Shepp [11], who showed that time-optimal paths for Hilare-like robots with actuator limits consist of straight-line segments and arcs of circles.

The nominal input for this trajectory is $u^0 = (1, 1)$. We set $\alpha = 0.1, \delta = 1.0$ as before. We again choose an initial condition error of (-0.1, 0.2, 0.1), and run the simulation for 2π seconds. See Figures (8 - 10) for the results.

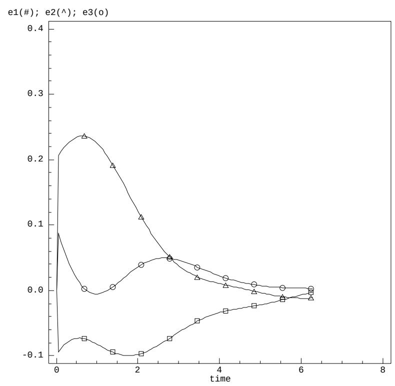

Figure 8. The errors e versus time.

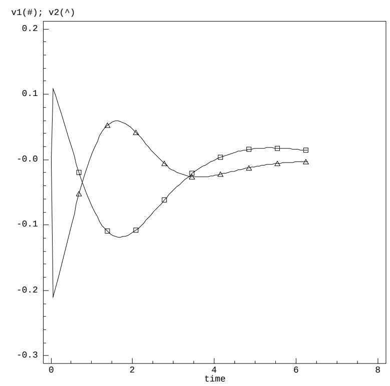

Figure 9. Error inputs v versus time.

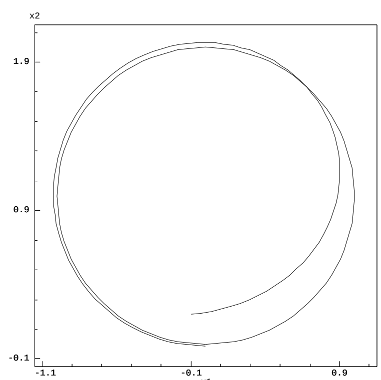

Figure 10: This phase plot shows the nominal and actual trajectories, projected onto the (x_1, x_2) plane (the orientation of the robot is not shown). The desired trajectory is the perfect circle. Note how quickly and smoothly the system converges to the desired trajectory.

Although we have used the same values of α, δ and initial error as in the previous example, the convergence seems less rapid, indicating that the convergence rate depends on the chosen trajectory. However, the convergence rate is also a

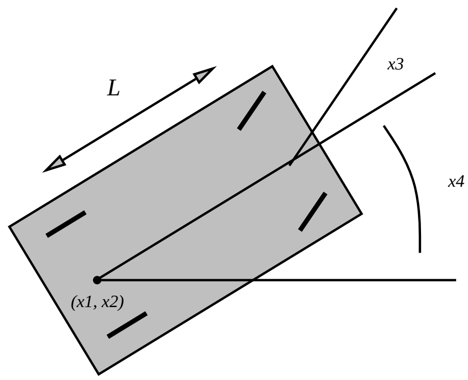

Figure 11: The front wheel drive car.

function of α, which we are free to choose. If we needed faster convergence, we could simply choose a larger α.

5 Trajectory Stabilization for a Front Wheel Drive Car

We consider now a front-wheel drive car. This system is also controllable [10], although two levels of Lie Brackets must be taken to show this. We quote here the kinematic equations, the reader interested in their derivation may consult [10]. A sketch of the car is found in Figure 11.

The system equations for the front wheel drive car (assuming velocities as inputs) are:

$$\begin{aligned}
\dot{x}_1 &= \cos(x_3)\cos(x_4)u_1 \\
\dot{x}_2 &= \cos(x_3)\sin(x_4)u_1 \\
\dot{x}_3 &= u_2 \\
\dot{x}_4 &= \frac{1}{\mathrm{L}}\sin(x_3)u_1 \qquad (12)
\end{aligned}$$

where (x_1, x_2) is the position of the car in the plane, x_3 is the angle of the front wheels with respect to the car (or the steering wheel angle), x_4 is the orientation of the car with respect to some reference frame, and the constant L is the length of the wheel base.

The matrices $A(t), B(t)$ are:

$$A(t) = \frac{\partial f}{\partial x}(x^0, u^0)$$

$$
= \begin{bmatrix} 0 & 0 & -\cos(x_4^0)\sin(x_3^0)u_1^0 & -\cos(x_3^0)\sin(x_4^0)u_1^0 \\ 0 & 0 & -\sin(x_3)\sin(x_4^0)u_1^0 & \cos(x_3^0)\sin(x_4^0)u_1^0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\mathrm{L}}\cos(x_3^0)u_1^0 & 0 \end{bmatrix}
$$

$$
B(t) = \begin{bmatrix} \cos(x_3^0)\cos(x_4^0) & 0 \\ \cos(x_3^0)\sin(x_4^0) & 0 \\ 0 & 1 \\ \frac{1}{\mathrm{L}}\sin(x_3^0) & 0 \end{bmatrix}
$$

Inspired by [10], we chose the nominal input $u^0 = (\sin(t), \cos(2t))$, roughly corresponding to a parallel- parking maneuver (see Figure 14). Again, we chose $\alpha = 0.1, \delta = 1.0$. After one period ($T = 2\pi$), this input steers the system in the direction given by the second-level nested Lie bracket of the two input vector fields (*i.e.* One write the system as $\dot{x} = g_1(x)u1 + g_2(x)u_2$, thus the nested Lie bracket would be $[g_1, [g_1, g_2]]$). Because the equations for this example are not simple, we have not tried to find H_c in closed form; all of the computations were done by the simulation program. The initial condition was chosen to be (0.1, -0.1, 0.05, 0.2), and the simulation was run for 2π seconds. Figures (12 - 14) show the results. Note the rapid convergence to zero in the error terms.

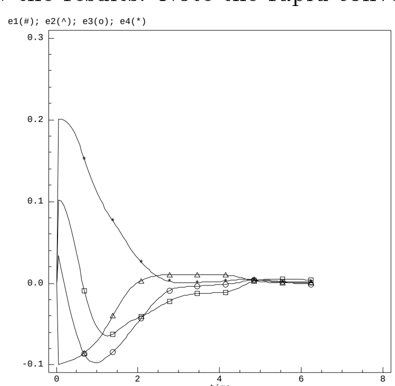

Figure 12: Plot of errors e versus time.

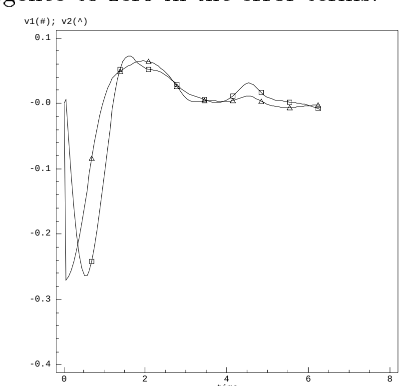

Figure 13: The control inputs v versus time. Note that they are bounded, smooth, and go to zero.

6 Conclusions

The control law and simulation results presented in this paper suggest that for nonholonomic systems, stabilizing to a trajectory is a better problem to consider than stabilizing to a point. It should be noted that for drift-free systems, *all* points are equilibrium points (in the sense that with zero input, the system will remain at rest).

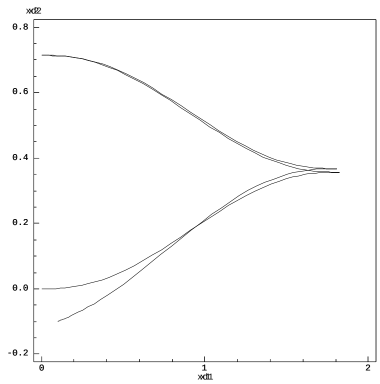

Figure 14: This phase plot shows the desired and the actual trajectories projected onto the (x_1, x_2) plane (the orientation of the car and the steering wheel angle are not shown). The desired trajectory is the one which starts at (0,0). Note how quickly and smoothly the control law stabilizes the system to this trajectory.

However, if one adopts our point of view, one must also face the problem of finding feasible trajectories; a rich problem which has not been solved for all systems. Excellent work has been done [7, 9, 10, 11, 13] in this area, and methods for finding trajectories exist for a wide range of nonholonomic systems (including all of the examples found in this paper).

In the context of non-kinematic models such as the real Hilare robot, the inputs are not the motor velocities but the torques, and thus the problems involving drift should also be examined. The control law presented here can be applied to stabilize systems with drift; however it is not as clear how to go about choosing the nominal trajectory. Further work could include the exploration and testing of other control laws for stabilizing linear time-varying systems.

The control law presented in this paper is robust to three types of error: initial condition errors, perturbations introduced along the trajectory, and noise in the sensor data. We have only shown the convergence results when there is an error in the initial condition, but it can be seen that the effects of the other two types of errors also are reduced using this law.

In summary, the path to a composite controller for mobile robots is: 1) Utilize the path planners to generate a nominal open-loop trajectory. 2) Apply the control law developed in this paper to stabilize the system to this nominal trajectory. 3) During operation of the robot, low-level sensor data can be used to avoid collisions caused by *a priori* errors in the knowledge of the environment. This new knowledge can be used to plan a new feasible nominal trajectory and find its associated stabilizing control law.

Acknowledgements

Part of this research was done at the Laboratoire d'Automatique et d'Analyse des Systèmes in Toulouse, France during the summer of 1991. G. Walsh, D. Tilbury, and S. Sastry are grateful for the warm welcome they received there. This research was supported in part by NSF grants ECS-87-19298 and IRI 90-14490, and the CNRS. The first author was also supported in part by an N.S.F. fellowship. The authors would like to thank Dr. Georges Giralt for many inspiring discussions and Andrew Teel for suggesting the two remarks after proposition one.

References

[1] G. Bastin and G. Campion. Feedback control of nonholonomic mechanical systems. *Advances in Robot Control*, 1991.

[2] R. W. Brockett. *Control Theory and singular Riemannian geometry.* Springer-Verlag, New York, 1981.

[3] R. W. Brockett. Asymptotic stability and feedback stabilization. In R.W. Brockett, R.S. Millman, and H.J. Sussman, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhauser, 1983.

[4] V. H. L. Cheng. A direct way to stabilize continuous-time and discrete-time linear time-varying systems. *IEEE Transactions on Automatic Control*, AC-24, No. 4:641 – 643, 1979.

[5] J. M. Coron. Gloabal asymptotic stabilization for systems without drift. Université de Paris, Sud, Orsay, 1991.

[6] C. Canudis de Witt and O. J. Sordalen. Exponential stabilization of mobile wheeled robots with nonholonomic constraints. In *IEEE Conference on Decision and Control*, pages 692 – 697, 1991.

[7] G. Lafferriere and H. J. Sussmann. Motion planning for controllable systems without drift: a preliminary report. Technical report, New York University and Rutgers University, June 1990.

[8] J-P. Laumond. Controllability of a multibody mobile robot. In *IEEE International Conference on Control and Applications*, pages 1033–1038, Pisa, Italy, 1991.

[9] J-P. Laumond and T. Siméon. Motion planning for a two degrees of freedom mobile robot with towing. In *IEEE International Conference on Control and Applications*, 1991.

[10] R. Murray and S. Sastry. Steering nonholonomic systems using sinusoids. In *IEEE Conference on Decision and Control*, pages 2097 – 2101, 1990.

[11] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2), 1990.

[12] Claude Samson. Feedback stabilization of a nonholonomic car-like mobile robot. In *IEEE Conference on Decision and Control*, 1991.

[13] D. Tilbury, J-P. Laumond, R. Murray, S. Sastry, , and G. Walsh. Steering car-like robots with trailers using sinusoids. In *IEEE International Conference on Robotics and Automation*, 1992.