# A Path Following Control for Unicycle Robots

**F. Díaz del Río,* G. Jiménez, J. L. Sevillano, S. Vicente, A. Civit Balcells**
*Robotics and Computer Technology for Rehabilitation Laboratory*

*Facultad de Informática. Universidad de Sevilla*
*Avda. Reina Mercedes s/n. 41012*
*Sevilla, Spain*

In this work we present a new path following control for unicycle robots that is applicable for almost all the possible desired paths and whose analysis is very straightforward. First we select the path following method that consists of two steps: choosing a ''projection'' that relates the actual posture to the desired path and imposing a ''motion exigency'' to ensure that the robot advances. A ''projection'' that considers all the error coordinates is selected and closed equations are obtained for it. The uniqueness projection is carefully analyzed and a necessary and sufficient condition is also presented. This condition shows that a slight bound on the curvature derivative of desired paths must be imposed to preserve uniqueness. It is remarkable that the selected path following is applicable for paths containing zero-radius turns, a problem that has never been resolved as far as we know. In addition, an asymptotically stable control law is found using the closed form equation of the proposed path following and the second Lyapunov method. Finally, we show the behavior of the path following and the control law through several simulated and experimental results, using a computerized wheelchair built at our research facility.

*To whom all correspondence should be addressed; e-mail: fdiaz@atc.us.es.
Contract grant sponsor: IASS agreement ''Segundo Convenio de Colaboración IASS-Universidad de Sevilla.''
Contract grant sponsor: CICYT.
Contract grant number: TER96-2056-C02-01.

## 1. INTRODUCTION

In the last few years there has been a great interest in finding controllers for nonholonomic mobile robots. The peculiar characteristics of the kinematic and dynamic model of these systems make them especially interesting. Moreover there is no doubt that their applications in the next few years will be large, in fields such as intelligent transportation systems, explorer vehicles, and personal or assistant robots.

Mobile robots are intrinsically nonlinear systems, because of their kinematic model. Moreover they have more state coordinates than degrees of freedom (DOF), because of their nonholomic constraints (except for the special case of omnidirectional robots). Due to this, convergence to a path may acquire a special treatment in mobile robots.

Many researchers have studied various tracking methods when the desired trajectory or path is memorized or previously generated. During the last years several methods have been proposed, and in a general sense we can distinguish between two main tracking methods. In a first group we find those that consider time explicitly in the tracking[1–6] (usually called *trajectory tracking*), and try to approach the robot to a moving point. In a second one, we find those that do not have timing requirements and try to converge to a path[7–14] (usually called *path following*). In the latter case the desired path is usually parameterized,[15] and the path following is identified with the progress of the descriptor parameter. Usually the parameter to describe a curve is the time, either the recorded time for previous trajectories or the real time when the tracking is done. However, many different parameters are possible. Note that in this case the time dependence is not relevant.

Trajectory tracking (TT) has been well studied because it is similar to servosystems, and it is guaranteed that the system will converge to the desired trajectory in a deterministic time using an asymptotically stable control law (except for the perturbations that it may suffer). On the other hand, path following (PF) is not well suited for systems with strict timing requirements, but it is very suitable for nonholonomic systems and is applicable to many mobile robots since they are not usually involved in hard real-time systems. Although the first approach seems to be the most straightforward, it has been shown that the second is more suitable for many situations in which time is not a critical parameter. This is the case for most applications in mobile industrial robots or assistant robots such as computerized wheelchairs. This situation can be understood if we consider the following example in TT systems: if big perturbations force the system to be at rest, the desired point for trajectory tracking will move unavoidably. This means that errors will grow up to some value that may introduce instability. On the other hand, if PF were used, the desired point will be the same despite these perturbations, because the path's shape and the real robot state remain the same. This allows the system to overcome large perturbations, avoiding possible unstable states. Thus interest in PF for mobile robots is rapidly growing.

Once a tracking method has been chosen, a convergence law must be found. Due to the special characteristics of the mobile robot state equations and the existence of several methods to track the reference, many researchers have investigated various ways to find a stable control law. Perhaps the most frequent contributions are those based on the Lyapunov direct method.[2,4,6] There have also been trials to linearize the system using a first-order approximation,[10,13] a local time-varying linearization around the equilibrium point,[16] or feedback linearization.[7,11] Sampei et al.[3] designed a controller using the exact linearization and time scale transformation. Recently, adaptive and learning control methods have been successfully applied to nonholonomic systems, including mobile robots.[17–19] Also conversion of these models into chained systems[20–22] has opened other possibilities for finding controllers for these systems. In addition, the techniques above can be mixed to obtain ''hybrid'' control laws; frequently these laws behave in a different manner according to the proximity to the target.[23,24] Finally we can find several excellent compendia in some reports or books.[7,25]

Our research group has been interested during the last few years in the improvement of electrical wheelchairs,[8,26,27] which incorporate advanced features. This field has interested many researchers in Europe during the last decade due to several projects that are trying to improve the quality of life of handicapped people.[28,29] Some features that should be incorporated into classical wheelchairs have been stated. In our group we have developed SIRIUS, an advanced wheelchair that includes path recovery of usual trajectories, detection and avoidance of obstacles through simple sensors like sonar, intelligent user interfaces with shared control, etc. Therefore SIRIUS can be considered to be a mobile robot that

can be teleoperated too. Discussing these aspects with trainers and users, we have concluded that playing back previously recorded trajectories is a very helpful aid. This avoids the user having to perform the difficult maneuvering of reverse driving and may be very useful in small areas like bathrooms.

One of the typical topologies for electric wheelchairs is the so-called unicycle or $(2,0)$-robot, according to the classification made by Campion et al.,[30] because their degree of maneuverability is 2 and their number of steering wheels is 0. They include driver motors at each rear wheel that can turn independently forward or backward. Different speeds at each rear wheel cause the turn of the chair. We have studied the complications that this topology introduces in path tracking, and we have discovered that the possibility of paths whose curvature tends to infinity (i.e., a zero-radius turn) is a very interesting problem for the path following method.

In the next sections we will try to analyze carefully the proposed PF construction and control law. In section 2 we define our robot model and we choose a set of coordinates, which are appropriate for describing a PF approach. In section 3, a PF construction is established based on two suggested steps: choosing a ''projection'' that relates the actual posture to the desired path, and imposing a ''motion exigency'' to ensure that robot advances. In section 4 we develop exhaustively the projection for SIRIUS, we find a closed form equation for the descriptor parameter of the desired path, we construct the projection on the absolute coordinate space, and we study the projection uniqueness. Once PF is well stated, we propose an asymptotically stable control law in section 5, which is evaluated through several simulated and real examples in section 6. Finally in section 7 we summarize the conclusions.

## 2. DEFINITIONS AND ROBOT MODEL

Let us consider the mobile robot shown in Fig. 1 (whose dimensions are those of SIRIUS) and let $\mathbf{q} = (X, Y, \phi)^t$ be its state coordinates, which represent the coordinates $(X, Y)$ of a certain point $\mathbf{P}_o$ (typically the midpoint between the rear wheels) on the basis of the fixed frame $(\Re) = (\mathbf{O}; \mathbf{i}, \mathbf{j})$ and the orientation $\phi$ of the robot with respect to the fixed frame. Other variables that characterize internal states, such as the angles turned by the wheels or

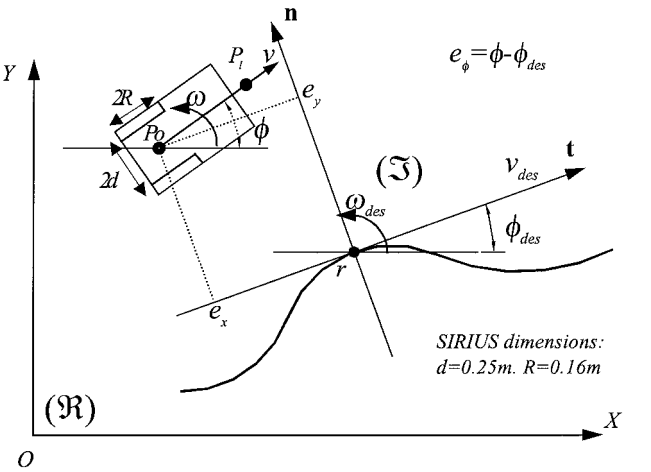


**Figure 1.** Extrinsic and intrinsic robot coordinates.

the relative orientation of castor wheels, do not represent any useful state for the tracking problem. The unicycle robot has three state variables but only 2 degrees of freedom as a result of the nonholonomic constraint. We assume that the wheels are nondeformable and that they are moving on a horizontal plane without slip to hold the constraint. Let $\mathbf{u} = (\nu, \omega)^t$ be the pair of input variables which represent the linear velocity of point $\mathbf{P}_o$ and the angular velocity of the robot, respectively.

For these vector variables the kinematic model of the unicycle robot can be expressed by the equations (that are nonlinear in $\mathbf{q}$ and linear in $\mathbf{u}$):

$$\dot{\mathbf{q}} = \mathbf{B}(\mathbf{q})\mathbf{u}; \qquad \mathbf{B} = \begin{bmatrix} \cos\phi & 0 \\ \sin\phi & 0 \\ 0 & 1 \end{bmatrix}; \qquad \mathbf{u} = \begin{pmatrix} \nu \\ \omega \end{pmatrix}; \qquad \mathbf{q} = \begin{pmatrix} X \\ Y \\ \phi \end{pmatrix} \tag{1}$$

In mobile robots the desired trajectory is usually recorded from a previous real trajectory or generated by a path generator module.[31] For our purposes both cases are the same, and the term *memorized path*, *reference path*, or merely *path* is used for both of them. For SIRIUS the usual path to be recovered is the previous path done by the user (usually actuating on the joystick), which must be repeated in reverse direction when the user gives this order.

A reference or desired path to be followed can be described by a single parameter, namely, $r$, and

it can be expressed as a vector of desired state coordinates $\mathbf{q}_{\text{des}}(r) = (X_{\text{des}}(r), Y_{\text{des}}(r), \phi_{\text{des}}(r))^t$.

To study the tracking of a memorized desired path $\mathbf{q}_{\text{des}}(r)$ let us define another intrinsic coordinate system $(\Im) = (\mathbf{q}_{\text{des}}(r); \mathbf{t}, \mathbf{n})$ linked to the path (usually called the Frenet frame[15]; see Fig. 1). $\mathbf{t}$ is the unitary vector parallel to the robot orientation in the desired point $\mathbf{q}_{\text{des}}$ and $\mathbf{n}$ the normal to it. Let $(e_x, e_y)$ be the position errors of point $\mathbf{P}_o$ relative to these axis and let $e_\phi$ be the robot orientation error, so $\mathbf{e}_q = (e_x, e_y, e_\phi)^t$ will be the relative errors vector (an analogous coordinate system was used by Kanayama et al.,[4] but their system was linked to the robot itself). Let $\mathbf{u}_{\text{des}}(r) = (\nu_{\text{des}}(r), \omega_{\text{des}}(r))^t$ be the desired inputs expressed as a function of the descriptor parameter $r$.

As we are interested in path following, the parameter $r$ will be chosen through some kind of relation between the actual system's state $\mathbf{q}$ and the memorized path. This relation will give us the desired point $\mathbf{q}_{\text{des}}$ and the way in which parameter $r$ varies in relation to $t$, i.e., a state equation for $r$. This will be obtained in the following sections.

Furthermore, applying the chain law for desired inputs, we can get to

$$\omega_{\text{des}}(t) = \frac{d\phi_{\text{des}}}{dt} = \frac{d\phi_{\text{des}}}{dr}\frac{dr}{dt} = \phi'_{\text{des}}(r)\dot{r} = \omega_{\text{des}}(r)\dot{r}$$

where $(')$ means differentiation with respect to $r$. The relation for $\nu_{\text{des}}$ is analogous if we consider the length $s_{\text{des}}$ of the path $(\nu_{\text{des}}(t) = (ds_{\text{des}}/dt))$. To sum up we can declare that

$$\mathbf{u}_{\text{des}}(t) = \mathbf{u}_{\text{des}}(r)\dot{r}$$

For the chosen frame, error coordinates $\mathbf{e}_q$ can be obtained as a rotation around an axis normal to the $XOY$ plane. In fact and according to Fig. 1:

$$\mathbf{e}_q = \mathbf{R}(\phi_{\text{des}})(\mathbf{q} - \mathbf{q}_{\text{des}});$$

$$\mathbf{R}(\phi_{\text{des}}) = \begin{pmatrix} \cos(\phi_{\text{des}}) & \sin(\phi_{\text{des}}) & 0 \\ -\sin(\phi_{\text{des}}) & \cos(\phi_{\text{des}}) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Therefore the general form of state equations for this kind of coordinates is

$$\dot{\mathbf{e}}_q = \mathbf{B}_{\text{des}}(\mathbf{e}_q)\mathbf{u}_{\text{des}} + \mathbf{B}(\mathbf{e}_q)\mathbf{u} \tag{2}$$

These equations are linear in all the input variables and nonlinear in state variables. In our case, using the above relative variables and coordinates linked to the path, and by simple calculations, the following state equations can be found[32]:

$$\begin{pmatrix} \dot{e}_x \\ \dot{e}_y \\ \dot{e}_\phi \end{pmatrix} = \begin{pmatrix} -\nu_{\text{des}} \\ 0 \\ -\omega_{\text{des}} \end{pmatrix} + \begin{pmatrix} 0 & \omega_{\text{des}} & 0 \\ -\omega_{\text{des}} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} e_x \\ e_y \\ e_\phi \end{pmatrix} + \begin{pmatrix} \cos(e_\phi) & 0 \\ \sin(e_\phi) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \nu \\ \omega \end{pmatrix} \tag{3}$$

This form agrees with the intuition that error variables must grow with both real and desired posture advancement.

Finally, as we are interested in convergence to a path, we will suppose in this work (unless otherwise stated) that the desired trajectory has no end. We do not allow the trajectory to end because convergence to a fixed point $\mathbf{q}_o$ cannot be achieved through a smooth feedback stabilization control law (a direct result of Brockett's theorem[33]).

## 3. PATH FOLLOWING CONSTRUCTION

### Previous Studies

During the last decade there has been a great research effort to develop a tracking based in path following. This has led to several good approaches that emphasized diverse aspects according to the particular characteristics of the analyzed system or the desired paths to be tracked. The most important can be summarized in the following categories:

- In a first category[2,7,13,20] the desired point in the path is obtained through a normal projection along the vector that we have called $\mathbf{n}$. Therefore this projection chooses the point of the desired path that has a null $e_x$ coordinate (see Fig. 1). Articles in this category impose a constant value for the variable $\nu$ to guarantee that the system always moves. Finally an asymptotically convergent control law is obtained and behavior for desired paths composed of circles and lines is shown through simulation. Paths containing circles with a small radius (we will call turns with a null radius and infinite curvature ''zero-radius turns'') are prohibited, so it is ensured that the normal projection exists and is unique.

- A similar path following was used in Navlab.[34] In this case, after finding the same normal projection point $\mathbf{P}_a$, the authors used a quintic spline to ''connect'' the actual robot posture with the desired posture on the point that is located at a certain distance $L$ far away from $\mathbf{P}_a$. As Navlab is a car-like robot, it cannot make zero-radius turns, so these paths were not considered. On the whole, the main drawback for the alternatives that build a fully specified curve between the actual state and the desired path and force the robot to follow this curve[26,34] is that the extraction of closed form equations is not easy, and thus the analytical proof of convergence is very difficult.
- In a similar approach[10] the desired posture is chosen as the point of the workspace path closest to the actual position $(X, Y)$. This approach has the same restrictions as the previous choice; in fact in the $XY$ plane, this projection coincides with the previous normal projection—a classical result of differential geometry.[35] In this work the proposed controller is designed only for straight-line and circular-arc paths to be tracked with a constant velocity.
- Another point of view for the projection[3] is to transform the kinematic equations of the mobile robot into a new time scale. In particular, the time scale is chosen to be identical to the distance along the desired path. However the desired paths are limited to straight lines, because the authors are concerned with the tracking of lines and the parking maneuver in a garage. The authors show that the new scale (the distance along the desired path) represents the desired posture obtained through the normal projection.
- In the last category[11] the projection point chosen by the authors is the one that minimizes the Euclidean distance between the real and the desired points $\mathbf{P}_l$ (see Fig. 1). Using point $\mathbf{P}_l$ they avoid paths with curvature tending to infinite. Again these authors obtain good convergence results for several paths (circles and lines) through a feedback linearization control law. But this strategy fails when the desired path is a turn around point $\mathbf{P}_o$. In this case any actual configuration (having different orientations) whose point $\mathbf{P}_l$ is *on* the desired position for $\mathbf{P}_l$ will have zero distance. That is, the couple $(X_l, Y_l)$ does not represent the whole state of a mobile robot, although these coordinates always change for every trajectory.

It is important to mention that previous studies focused on some particular shapes of the desired paths. In opposition, in SIRIUS we must contemplate all the possible desired paths that can be made by the user (usually driving his or her joystick), including zero-radius turns. This is why we use a different path following approach, first proposed in ref. 8 and continued in refs. 32 and 9.

While trajectory tracking construction is elementary (it requires only choosing the most suitable relation $r = r(t)$), path following construction is not so simple because it implies some special relationship between the actual point and the global desired path, and between the inputs $\mathbf{u}$. In this work we propose a PF construction based on two steps (Fig. 2). In this figure we begin with a unicycle robot that has three state coordinates (three error coordinates $\mathbf{e}_q$ expressed relative to the desired path) and 2 (DOF)$\mathbf{u}$.

### First Step: Projection

The projection $f_{\text{proj}}(\mathbf{q}, r) = 0$ or $f_{\text{proj}}(\mathbf{e}_q, r) = 0$ relates real posture with the desired path (the dependence on the desired path can be condensed on the descriptor parameter $r$). This gives us a projecting point on the desired path: it is the desired posture $\mathbf{q}_{\text{des}}(r(t))$ at this instant of time. This projection may also be expressed as $f_{\text{proj}}(\mathbf{e}_q, r) = 0$.

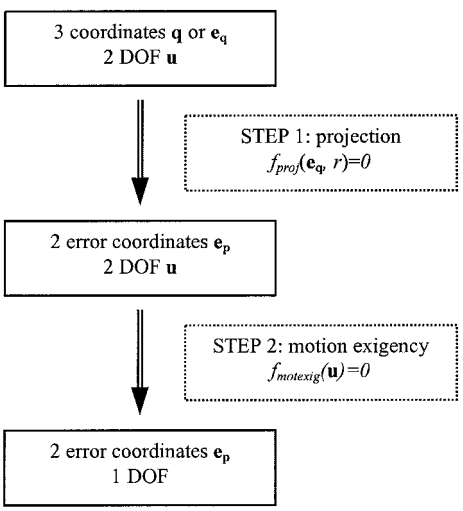


**Figure 2.** General path following construction scheme.

The projection is also a holonomic constraint between error coordinates; thus it implies the elimination of one state coordinate. Hence the robot posture will now be given by only two error coordinates plus the parameter $r$ (that provides all the information about the desired posture, e.g., the reference frame), instead of the three $\mathbf{e}_q = (e_x, e_y, e_\phi)$ (Fig. 3). We will call the new error coordinates $\mathbf{e}_p = (e_1, e_2)$. Points $\{\mathbf{e}_q\}$ that obey $f_{\text{proj}}(\mathbf{e}_q, r) = 0$ define a surface (two-dimensional in our case) where the robot is placed. Vector $\mathbf{u}$ cannot play a role in this step, because we are talking about a geometric projection.

Uniqueness of the projected point on the path $\mathbf{q}_{\text{des}}(r)$ is not always guaranteed, but the inversion of the projection must be ensured at least locally, so that the convergence can be proved in a neighborhood of the desired path. This problem will be discussed extensively in the next section.

A classical example of one of these projections is the normal projection,[2,7,20] equivalent to making $e_x$ null, i.e.,

$$f_{\text{proj}}(\mathbf{e}_q, r) = 0: \qquad e_x = 0$$

That is, the first error coordinate $e_x$ is eliminated and the robot posture is expressed by only two coordinates: $\mathbf{e}_p = (e_y, e_\phi)$ [that are called $(y, \theta)$ in these references]. The two-dimensional surface is the $e_y$ axis extended for all the possible robot orientations.

It is important to remark that parameter $r$ is now the third state coordinate (in trajectory tracking $r$ gives us no state because $r$ is determined only by time), and we should include it to specify the whole robot posture, now given by $(r, e_1, e_2)$. In this work we do not consider $r$ as an *error coordinate*, because it does not play any role in our stabilization problem (it will only be focused on making $e_p \to 0$, regardless of $r$).

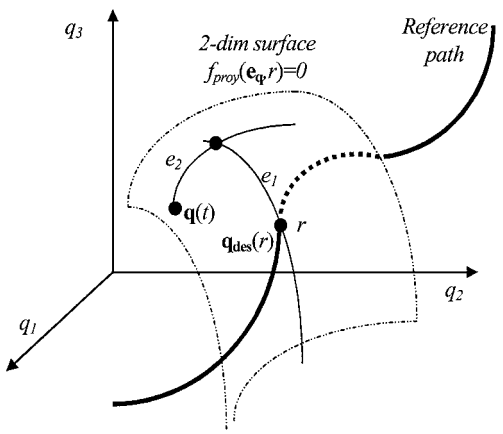


**Figure 3.** Coordinate transformation due to the projection.

Differentiating the projection $f_{\text{proj}}(\mathbf{e}_q, r) = 0$ we obtain the way in which parameter $r$ varies (an equation for $\dot{r}$):

$$\frac{\partial f_{\text{proj}}}{\partial \mathbf{e}_q}\dot{\mathbf{e}}_q + \frac{\partial f_{\text{proj}}}{\partial r}\dot{r} = 0$$

Now state Eq. (2) can be substituted, and solving for $\dot{r}$, we finally have

$$\dot{r} = \frac{\dfrac{\partial f_{\text{proj}}}{\partial \mathbf{e}_q}\mathbf{B}(\mathbf{e}_q, r)\mathbf{u}}{\dfrac{\partial f_{\text{proj}}}{\partial r} + \dfrac{\partial f_{\text{proj}}}{\partial \mathbf{e}_q}\mathbf{B}_{\text{des}}(\mathbf{e}_q)\mathbf{u}_{\text{des}}} \tag{4}$$

If the denominator is null in Eq. (4), the variation of $r$ is undefined. In the next section we will see that this case is equivalent to the non-uniqueness of the chosen projection.

A straightforward example of the $\dot{r}$ equation is the normal projection mentioned before. If we differentiate this projection we get

$$\dot{e}_x = 0 \Rightarrow -\nu_{\text{des}} + \omega_{\text{des}} e_y = \nu \cos(e_\phi)$$

Using the chain law for desired input $\mathbf{u}_{\text{des}}(t) = \mathbf{u}_{\text{des}}(r)\dot{r}$, we finally obtain

$$-\nu_{\text{des}}(r)\dot{r} + \omega_{\text{des}}(r)\dot{r}e_y = \nu\cos(e_\phi)$$

$$\Rightarrow \dot{r} = \frac{\nu\cos(e_\phi)}{\nu_{\text{des}}(r) - \omega_{\text{des}}(r)e_y}$$

This is the same equation for the normal projection, expressed there using the natural arc parameter $s$, which makes $\nu_{\text{des}}(s) = 1$ and $\omega_{\text{des}}(s) = C_c(s)$, the planar curvature.

### Second Step: "Motion Exigency"

Finally, we need to impose a ''motion exigency'' to guarantee that the robot moves. Its form and its variation with time depend on the robot topology and even on the application. For example, in car-like robots a simple and adequate exigency (usually

found in the literature) is $\nu = \text{constant}$. This selection is based on the assumption that angular velocities are never big in cars, and thus linear velocity $\nu$ is ''eliminated'' and angular velocity is actually the only DOF to converge to a path.

However this is not the only suitable motion exigency for mobile robots; for example, another motion exigency used in the literature,[3] is that obtained for a constant centrifugal force. The resultant exigencies are hyperbolas in the $\mathbf{u}$ plane, with a discontinuity in the axis that must be avoided.

SIRIUS motors do not have motion restrictions. Thus the angular and linear velocities may be equivalent since both increase linearly with the independent velocities of the drive wheels $(\dot{\theta}_R, \dot{\theta}_L)$, according to the expression

$$\begin{pmatrix} \nu \\ \omega \end{pmatrix} = \mathbf{D} \begin{pmatrix} \dot{\theta}_R \\ \dot{\theta}_L \end{pmatrix}; \qquad \mathbf{D} = \begin{bmatrix} \frac{R}{2} & \frac{R}{2} \\ \frac{R}{2d} & -\frac{R}{2d} \end{bmatrix}$$

Furthermore, the desired trajectories made by a user in domestic environments contain indistinctly paths with very low or very high planar curvature (i.e., $\omega_{\text{des}}/\nu_{\text{des}}$), due to the narrow areas where wheelchairs must move. Then a very adequate motion exigency is a function that splits the whole motion symmetrically between $\nu$ and $\omega$, i.e., the ellipse

$$f_{\text{motexig}}(\mathbf{u}) = 0 \Rightarrow \nu^2 + b_\phi^2 \omega^2 = K_{mov}^2 > 0 \qquad (5)$$

where $K_{\text{mov}}$ is the whole motion applied (that may vary with time or as a function of several factors according to the specific application). The parameter $b_\phi$ is a scale factor to guarantee dimensional homogeneity. Besides, it indicates the degree of turning that we want the robot to perform: i.e., if $b_\phi$ is low then the robot will turn slowly; however if it is high, the motion exigency should ask the robot for a faster rotation. Therefore $K_{\text{mov}}$ limits the maximum linear velocity and $K_{\text{mov}}/b_\phi$ is the maximum angular velocity.

From the practical point of view, slippage avoidance must be imposed to the collected trajectories; i.e., violent movements of SIRIUS cannot be allowed if the user wants to recover a trajectory. Slippage is more likely when angular velocity is high, so this velocity is bounded on paths done by the user (i.e., $\omega_{\text{des}}$), and for the same reason, the real angular velocity when following a memorized path (i.e., $\omega$) must also be bounded. This is another role that constant $b_\phi$ will play in the motion exigency. This bound does not limit the set of feasible desired paths or the convergence to them.

Now that we have proposed a PF construction, we will apply it thoroughly to the case of SIRIUS in the next section.

## 4. ANALYSIS OF THE PATH FOLLOWING PROJECTION FOR SIRIUS

### Projection Selection

SIRIUS belongs to the group of (2, 0)-robots,[30] one of the most usual mobile robot configurations. Its trajectories are complicated, because it cannot have complete maneuverability (that is, it is not omnidirectional), but it can make zero-radius turns. Thus it is a very interesting problem to find a suitable path following projection for these robots. Moreover the uniqueness of the projection must be deeply analyzed to find a condition that ensures its completion.

In some applications only a set of state coordinates is needed for the path following (e.g., if only lines must be followed). In SIRIUS, due to earlier reasons we have to consider the whole robot state. Furthermore we should consider the three coordinates in a similar fashion as long as maneuverability in these robots is very high and they have no additional movement restrictions. Therefore we choose the point in the path nearest to the robot as the projecting point, i.e., the one for which distance is minimal, and a ''good'' alternative for the distance $d_q$ is

$$d_q^2 = e_x^2 + e_y^2 + K_\phi^2 e_\phi^2 \qquad (6)$$

where $K_\phi$ is a scale factor to guarantee dimensional homogeneity. Another advantage of this selection is that it will ensure uniqueness in a tube around a desired path if a slight bound on the desired paths is preserved.

Errors in this distance can vary because of two reasons: real robot movement (i.e., because of $\mathbf{u}$) and selection of projecting point (i.e., variation or $r$). To choose the minimal distance point on the desired path we ''freeze'' actual robot posture ($\mathbf{u} = 0$) and ''move'' along the desired path ($r$ is varied) looking

for the point with a local minimum

$$\left.\frac{\partial d_q^2}{\partial r}\right|_{\mathbf{u}=0} = 0 \Rightarrow 2e_x e'_x + 2e_y e'_y + 2K_\phi^2 e_\phi e'_\phi|_{\mathbf{u}=0} = 0$$

where $(')$ holds for differentiation with respect to $r$ and $(\dot{})$ with respect to $t$. Now we must substitute the state equations, expressed with $\mathbf{u} = 0$, and considering only the variation of $r$ (time is ''frozen''), that is,

$$\begin{pmatrix} e'_x \\ e'_y \\ e'_\phi \end{pmatrix} = \begin{pmatrix} -\nu_{\text{des}}(r) + \omega_{\text{des}}(r)e_y \\ -\omega_{\text{des}}(r)e_x \\ -\omega_{\text{des}}(r) \end{pmatrix}$$

Substituting in the state equations and simplifying we finally have the projection:

$$e_x \nu_{\text{des}}(r) = -K_\phi^2 e_\phi \omega_{\text{des}}(r) \tag{7}$$

As our projection function chooses the desired point nearest to the path, its behavior should be intuitive. This can be completely shown[32] through several examples. In this article we have selected two simple examples that show the projection behavior in extreme cases, namely, when the desired path is a straight line and when it is a zero-radius turn (Fig. 4).

In the first case $\nu_{\text{des}}(r) \neq 0$ and $\omega_{\text{des}}(r) = 0$, and thus the projection simplifies to $e_x = 0$; i.e., it coincides with the classical normal projection. Therefore when the desired path is a ''pure advance,'' the projection gives the maximum relevance to the coordinate directly linked to the linear velocity, i.e., $e_x$. On the other hand, in the second case $\nu_{\text{des}}(r) = 0$ and $\omega_{\text{des}}(r) \neq 0$, and the projection simplifies to $e_\phi = 0$ (normal projection[2,7,12,20] is not defined in this case); i.e., when desired path is a ''pure turn,'' the maximum relevance is given to the orientation.

Another interesting consequence is that projection problems disappear when the path is a circle and the robot is in the center. Normal projection is undefined in this case. Conversely, using the minimal distance projection, the projecting point will be just the desired point with the same orientation or $e_\phi = 0$ (in the center $e_y$ is constant and $e_x$ is null for every desired point). Moreover we could get the normal projection as a particular case, for example, by taking $K_\phi = 0$ and supposing that $\nu_{\text{des}}(r) \neq 0$, because normal projection to a planar curve coincides with minimal Euclidean distance.[35]

## State Equation for the Descriptor Parameter

As we described in the previous section, differentiation of the projection gives us the new state equation for parameter $r$. Differentiating the projection with respect to time, and substituting the state equations (we express explicitly the dependence of the desired inputs, to distinguish between dependence on $r$ or $t$):

$$\begin{aligned} &\left[-\nu_{\text{des}}(t) + \omega_{\text{des}}(t)e_y + \nu\cos(e_\phi)\right]\nu_{\text{des}}(r) \\ &\quad + e_x \nu'_{\text{des}}(r)\dot{r} \\ &\quad = -K_\phi^2[\omega - \omega_{\text{des}}(t)]\omega_{\text{des}}(r) - K_\phi^2 e_\phi \omega'_{\text{des}}(r)\dot{r} \end{aligned}$$

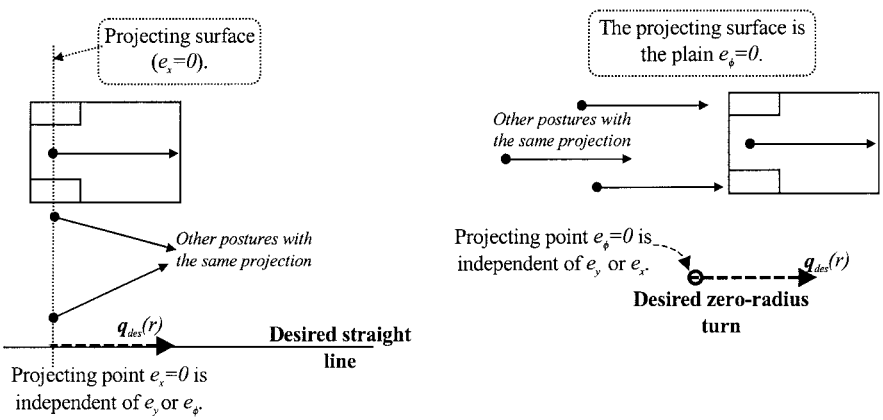


**Figure 4.** Projection behavior for a line and for a zero-radius turn.

Finally using the chain law, and solving for $\dot{r}$:

$$\dot{r} = \frac{\nu\nu_{\text{des}}(r)\cos(e_\phi) + K_\phi^2\,\omega\omega_{\text{des}}(r)}{\nu_{\text{des}}^2(r) - \omega_{\text{des}}(r)\nu_{\text{des}}(r)e_y + K_\phi^2\,\omega_{\text{des}}^2(r) - K_\phi^2 e_\phi\,\omega'_{\text{des}}(r) - e_x\,\nu'_{\text{des}}(r)} \tag{8}$$

where we have supposed that the denominator of this equation is not null (we will see below that denominator nullity is directly related to non-uniqueness). Here $\nu$ and $\omega$ are the real inputs, but $\nu_{\text{des}}(r)$ and $\omega_{\text{des}}(r)$ are the input profiles that describe the desired path.

We have to remark that this equation seems to contradict the path following construction since it uses the three error coordinates $(e_x, e_y, e_\phi)$. Nevertheless only two of the three errors are independent, and the third one is completely determined by the projection. In the rest of this article we will use the three errors and the parameter $r$ state equations *plus* the projection, always bearing in mind that the projection eliminates one of the errors.

Previous problems have arisen from the selection of a coordinate system $(e_x, e_y, e_\phi)$ relative to the desired path, whose state equations are quite simple. In any case, a coordinate system in which the robot's state can be represented only by two independent errors can be found. This can be achieved, for example, if we represent the robot trajectories in the three-dimensional configuration space defined by $(X, Y, \lambda\phi)$, where the constant $\lambda$ (homogeneous to a distance) is introduced to ensure dimensional homogeneity. In this space we can define (as usual in differential geometry[15]) the Frenet trihedron $\Im$ associated to a point in the desired path $\mathbf{q}_{\text{des}}$. It can be demonstrated that the projection on the normal plane (defined by the normal and binormal vectors[15]) will give us a representation with only two error coordinates. We can name them $\mathbf{e}_p = (e_n, e_b)$, and the actual posture is then expressed by $\mathbf{e}_p$ plus the chosen parameter $r$. This procedure is fully detailed in ref. 32, and finally we get a very complicated set of state equations. These equations permit us to obtain a driftless representation of the system.

Therefore the PF method reduces the system to a driftless system with two errors $\mathbf{e}_p$ and the parameter $r$, according to the method shown in Fig. 2:

$$\begin{aligned} \dot{r} &= B_r(r, \mathbf{e}_p)\mathbf{u} \\ \dot{\mathbf{e}}_p &= \mathbf{B}_e(r, \mathbf{e}_p)\mathbf{u} \end{aligned} \tag{9}$$

Regardless of their generality, these three-dimensional configuration space equations are not useful in practice mainly for the following reasons:

- A lot of off-line calculations must be done to express the Frenet's vectors and other magnitudes as a function of the chosen parameter.
- On-line integration of the resultant state equations is far more complex than that resulting from using the above $(\dot{r}, \dot{e}_x, \dot{e}_y, \dot{e}_\phi)$ state equations, despite the fact that the last one needs one additional integration.
- Moreover, the previous calculations may accumulate numeric errors that may be important if complicated magnitudes are involved in the equations.

**Table I.** Different cases for different desired path curvatures.

| Case | Resultant projection | Variation of parameter $r$ | Possible pair |
|---|---|---|---|
| 1. Line: $\omega_{\text{des}} = 0$ $e_x = 0$ | $e_x = 0$ | $\dot{r} = \dfrac{\nu\cos e_\phi}{\nu_{des}(r)}$ | $(e_y, e_\phi)$ |
| 2. ''Pure turn'': $\omega_{\text{des}} = 0$ and $\nu_{\text{des}} = 0$ | $e_\phi = 0$ | $\dot{r} = \dfrac{\omega}{\omega_{des}(r)}$ | $(e_x, e_y)$ |
| 3. Circle with curvature: $K_{\text{des}} = \omega_{\text{des}}/\nu_{\text{des}}$ | $e_x = -K_\phi^2 K_{\text{des}} e_\phi$ | $\dot{r} = \dfrac{\nu\cos e_\phi + K_\phi^2\,\omega K_{\text{des}}(r)}{\nu_{\text{des}}(r) - K_{\text{des}}(r)\nu_{\text{des}}(r)e_y + K_\phi^2 K_{\text{des}}^2(r)\nu_{\text{des}}(r) - K_\phi^2 e_\phi K'_{\text{des}}(r)}$ | $(e_y, e_\phi)$ $(e_x, e_y)$ |

In conclusion, these reasons make it preferable to construct the path following using the three errors $(e_x, e_y, e_\phi)$, but always taking into account the fact that the projection eliminates one of them.

Besides, for some simple paths we could be able to use an explicit pair of coordinates (Table I). In this table we can see also that the form of $\dot{r}$ for these simple cases agrees with what could be expected. For example, if the desired path is a line, $r$ will only grow by means of the component parallel to that line $(\nu \cos e_\phi)$, and $e_x$ does not express any state because it is suppressed by the projection. Analogously, if the desired path is a zero-radius turn, $r$ grows by means of $\omega$, and $e_\phi$ is not a valid coordinate. Magnitudes $\nu_{\text{des}}(r)$ and $\omega_{\text{des}}(r)$ in the denominator play the role of a scale factor since the description of the path is linked to the way in which $r$ varies. For a circle we could work either with $(e_y, e_\phi)$ or $(e_x, e_y)$ indistinctly; the third coordinate would be calculated through the projection. In conclusion and as was described above, we will use in the rest of the article these three errors plus the projection, to ensure generality for all the possible reference paths.

Another problem is the inversion of the projection (i.e., the uniqueness of the projecting point). If the desired path is a straight line or a pure rotation, it is easy to see that the inversion is guaranteed. In most complex paths, the inversion must be carefully analyzed, so we will dedicate the following section to this problem.

## Projection Uniqueness

The verification of projection uniqueness depends heavily on the projection form. A way to study uniqueness breakage is by finding the paths or the pieces of path that fulfill the chosen projection continuously, i.e., a interval $]r_1, r_2[$ with the following property:

$$\exists \mathbf{e}_q : \forall r \in ]r_1, r_2[ \, , \, f_{\text{proj}}(\mathbf{e}_q, r) = 0$$

We will call these pieces of path *equiprojecting* or *equidistant* segments, since the projection chooses the desired posture with minimal distance.

Some of the *equiprojecting* segments can be easily found; for example, in the classical normal projection one can quickly notice that a circle centered on actual posture $\mathbf{q}_o$ breaks local uniqueness, since all the normal vectors to each circle point pass through its center. Nevertheless, the following theorem gives us a necessary and sufficient condition to find equiprojecting segments for any projection and any fixed real posture $\mathbf{q}_o$.

**Theorem 1:** *Projection uniqueness is broken for an interval* $]r_1, r_2[$ *if and only if we have*:

1. *The denominator of the* $\dot{r}$ *Eq.* (4) *is null.*
2. $\exists r_o \in ]r_1, r_2[ | f_{\text{proj}}(\mathbf{e}_q, r_o) = 0$

*Proof: Necessary condition*. Let us make the inputs $\mathbf{u} = 0$ to fix the real posture $\mathbf{q} = \mathbf{q}_o$ and find all the possible equiprojecting segments around it through the variation of parameter $r$. Therefore, let us consider $f_{\text{proj}}(\mathbf{e}_q, r)$ to be a function of $r$. The segment described by $]r_1, r_2[$ will be equiprojecting if this function is null over $]r_1, r_2[$. That is, at least this function is zero at one of these points (condition 2) and it is constant over it. This last condition is true if

$$\frac{df_{\text{proj}}\left(\mathbf{e}_q(r), r\right)}{dr} = 0 \Rightarrow \frac{\partial f_{\text{proj}}}{\partial \mathbf{e}_q} \frac{d\mathbf{e}_q}{dr} + \frac{\partial f_{\text{proj}}}{\partial r} = 0$$

In addition state Eqs. (2) for null inputs reduce to $\dot{\mathbf{e}}_q = \mathbf{B}_{\text{des}}(\mathbf{e}_q)\mathbf{u}_{\text{des}}(r)\dot{r}$. Applying the chain law we have that for $\mathbf{u} = 0$ $\dot{\mathbf{e}}_q = (d\mathbf{e}_q/dr)\dot{r}$, so the term $d\mathbf{e}_q/dr$ equates to $\mathbf{B}_{\text{des}}(\mathbf{e}_q)\mathbf{u}_{\text{des}}(r)$, which, when substituted in the previous equation, gives us the first condition.

*Sufficient condition*. Similarly if we have that a piece of path described by $]r_1, r_2[$ fulfills conditions 1 and 2 at an instant $t_o$, from condition 2 we have that one of its points is a projecting point for the actual posture $\mathbf{q}(t_o)$ [i.e. $f_{\text{proj}}(\mathbf{e}_q, r_o) = 0$]. Moreover for this particular instant and over the interval $]r_1, r_2[$, we know (from condition 1) that

$$\frac{\partial f_{\text{proj}}}{\partial \mathbf{e}_q} \frac{d\mathbf{e}_q}{dr} + \frac{\partial f_{\text{proj}}}{\partial r} = 0$$

which means that $f_{\text{proj}}$ is a constant with respect to $r$. Therefore $f_{\text{proj}}(r) = \text{constant} = 0$; i.e., this segment is equidistant. ■

An interesting point in this discussion is to demonstrate whether a projection whose uniqueness is guaranteed for all feasible paths can exist. We leave this for further studies because the minimal distance projection is enough for our system SIRIUS.

Let us apply the previous theorem to our projection to find possible equidistant segments. This

leads us to the solution of the differential equation

$$\nu_{\text{des}}^2(r) - \omega_{\text{des}}(r)\nu_{\text{des}}(r)e_y + K_\phi^2\omega_{\text{des}}^2(r) - K_\phi^2 e_\phi \omega'_{\text{des}}(r) - e_x \nu'_{\text{des}}(r) = 0$$

Equation (10) has no solution if $\omega_{\text{des}}$ or $\nu_{\text{des}}$ is null. Therefore we can define $K_{\text{des}}(r) = (\omega_{\text{des}}(r)/\nu_{\text{des}}(r))$, and $K_{\text{des}}(r) \neq 0$ as well. $K_{\text{des}}(r)$ represents the curvature on plane $XOY$ of the equidistant segment that we are looking for, and thus (see the denominator of case 3 in Table I) the previous equation is

$$1 - K_{\text{des}}(r)e_y + K_\phi^2 K_{\text{des}}^2(r) - \frac{K_\phi^2 e_\phi K'_{\text{des}}(r)}{\nu_{\text{des}}(r)} = 0 \quad (10)$$

Let us suppose that an equidistant segment is described with linear constant speed equal to 1; that is, $\nu_{\text{des}}(r) = 1$, $r = s_{\text{arc}}$ is the natural arc parameter[15] on plane $XOY$. Let us express this equation as a function of errors $e_y, e_\phi$ leading to a line equation in the errors plane (it is not necessary for this plane to take into account the third error $e_x$ since it is readily given by the projection):

$$e_y = me_\phi + b$$
$$m = -\frac{K_\phi^2 K'_{\text{des}}}{K_{\text{des}}}; \qquad b = \frac{1 + K_\phi^2 K_{\text{des}}^2}{K_{\text{des}}} \quad (11)$$

Now let us suppose that the control law maintains the robot below certain maximum errors, for example, in an ellipse defined by a bigger radius of $\varepsilon$ and an eccentricity of $\delta$ (a similar development can be made for a maximum errors rectangle in the errors plane). For SIRIUS the maximum errors allowed cannot be very high because it navigates in a narrow environment (e.g., the user's home); in fact, $e_{x,\max} = 0.1$ m, $e_{y,\max} = 0.1$ m, and $e_{\phi,\max} = 0.2$ rad. Obviously bigger errors would not correspond to a practical situation, i.e., the control system would probably have failed or the tracking would not be working properly. Then a possible equidistant segment will exist if there is a solution for the system

$$e_y = me_\phi + b$$
$$e_y^2 + (\delta K_\phi e_\phi)^2 = \varepsilon^2 \quad (12)$$

To find a possible solution, we can substitute $e_y$ in the second equation of (12) and obtain a second-order equation for $e_\phi$:

$$e_\phi^2\left(m^2 + \delta^2 K_\phi^2\right) + 2mbe_\phi + b^2 - \varepsilon^2 = 0$$

The discriminant of the previous equation is

$$\Delta = (2mb)^2 - 4(b^2 - \varepsilon^2)\left(m^2 + \delta^2 K_\phi^2\right)$$

Then the second-degree equation will have no solution if its discriminant is negative. Therefore substituting the values of Eq. (11):

$$-\left(\frac{1 + K_\phi^2 K_{\text{des},o}^2}{K_{\text{des},o}}\right)^2 \delta^2 K_\phi^2 + \left(\frac{-K_\phi^2 K'_{\text{des},o}}{K_{\text{des},o}}\right)^2 \varepsilon^2 + \varepsilon^2\delta^2 K_\phi^2 < 0 \Leftrightarrow (K'_{\text{des},o})^2 < \frac{\delta^2}{K_\phi^2}\left[\left(\frac{1 + K_\phi^2 K_{\text{des},o}^2}{\varepsilon}\right)^2 - K_{\text{des},o}^2\right]$$

Finally this last condition leads us to a bound to avoid equidistant segments, here expressed in terms of $K'_{\text{des}}$:

$$(K'_{\text{des}})^2 = \left(\frac{dK_{\text{des}}(s_{\text{arc}})}{ds_{\text{arc}}}\right)^2 < \frac{\delta^2 K_{\text{des}}^2}{K_\phi^2}\left[\left(\frac{1 + K_\phi^2 K_{\text{des}}^2}{\varepsilon K_{\text{des}}}\right)^2 - 1\right] = f_{\text{bound}}^2(K_{\text{des}}) \quad (13)$$

As a result, this bound (13) is imposed in SIRIUS when the user is doing a trajectory that he or she wants to recover (in reverse direction), thus avoiding the existence of equiprojecting segments. Obviously this bound limits the set of feasible paths, but the limitation is minimal and when driving the joystick this bound is never exceeded, because these paths are not usual. Note that if the curvature is zero, the bound of $K'_{\text{des}}$ is very high, and if curvature is large, then the bound function tends to $K_{\text{des}}^2$, multiplied by some constant factor.

In SIRIUS we apply another simplified bound that is a little more restrictive:

$$|K'_{\text{des}}| < \frac{\delta}{\varepsilon K_\phi}\left(1 + 0.81K_\phi^2 K_{\text{des}}^2\right) \leq f_{\text{bound}}(K_{\text{des}})$$

Integration (for example, in $s_{\text{arc}} > 0$) of this new bound gives an idea of the less abrupt path that would transgress the bound:

$$K_{\text{des}}(s_{\text{arc}}) = \frac{1}{0.9K_\phi} \tan\left(\frac{0.9\delta}{\varepsilon} s_{\text{arc}}\right)$$

For SIRIUS $\varepsilon^2 = 0.01$ m$^2$ and $\delta = 2$ (with $K_\phi =$ 0.25 m; a lower $K_\phi$ value is not practical as shown in the next paragraph). Therefore the desired path should change the robot trajectory from a straight line to a small circle of less than 1 cm radius (almost a zero-radius turn) in less than 6 cm of advance, which will not be found in an ordinary user's path. Instead of those brusque paths, the user usually stops the chair and begins a new segment with high curvatures (thus changing from a line to an almost zero-radius turn). The case of paths composed of segments is frequent in teleoperated navigation and it is discussed below.

We can obtain a lower bound for $K_\phi$ as it is related to constant $b$. Higher absolute values of $b$ reduce the possibility of equidistant segments since an increase in $b$ separates the line from the ellipse. The worst case corresponding to the minimum of $b$ happens at $K_{\text{des}} = 1/K_\phi$ where $b = 2K_\phi$. As constant $b$ represents the value of $e_y$ for null $e_\phi$, we should ensure that $b$ is far from $e_{y,\max}$ or equivalently $2K_\phi$ is far from $e_{y,\max}$.

In conclusion, the previous bound is a condition to preserve projection uniqueness. Uniqueness will be preserved particularly if errors are maintained far from line (11); i.e., errors are maintained on a tube around the path whose width is variable in function of the desired $K_{\text{des}}$ and $K'_{\text{des}}$. In SIRIUS we prefer to impose this condition over the collected paths (since it does not restrict user's paths severely) rather than modify the desired path in any way, because the last may cause a collision in the recovery.

Finally in this discussion we have not included paths made of fragments that contain desired points with null inputs $\mathbf{u}_{\text{des}}$ (i.e., fixed points). The reason is that a geometric projection is not properly defined if a desired path is finite. To solve this problem in SIRIUS, the desired fixed points (that divide a trajectory in two pieces) are detected when the user's paths are collected and especially labeled in the whole memorized path. When SIRIUS is near to a fixed point $\mathbf{q}_{\text{des}}(r_o)$ a ''jump'' from the projection on the first piece (desired point 1 on Fig. 5) to the second (desired point 2 on Fig. 5) must be done.

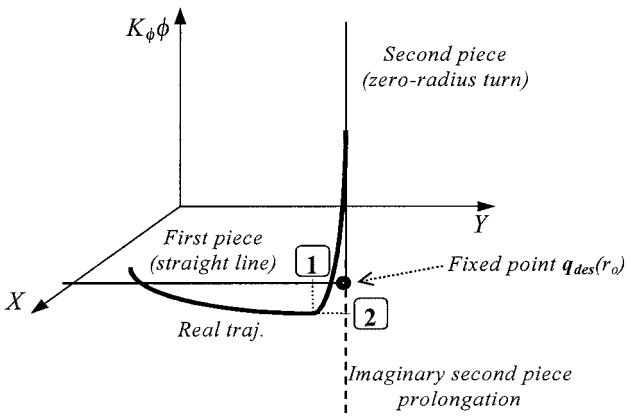


**Figure 5.** Projection on a path composed of two pieces.

Simultaneously the motion exigency is made lower [$K_{\text{mov}}$ of (5) decreases], to ensure that the robot is stopping. Therefore when $K_{\text{mov}}$ is sufficiently low and SIRIUS is close enough to the fixed point, the projection on the second piece is found. Note that errors relative to the first piece can be expressed like absolute errors with respect to the fixed point $\mathbf{q}_{\text{des}}(r_o)$ that is common to both pieces. Moreover the second piece is prolonged if it is necessary with an imaginary constant curvature piece to find the projection on it. The decrease of $K_{\text{mov}}$ and the moment where the ''jump'' is done are adjusted empirically.

## 5. LYAPUNOV-BASED CONTROL LAW

In this section we propose an asymptotically stable control law to converge to paths using Lyapunov's second method. Moreover the control law is globally asymptotically stable on a ball of fixed size given by $d_q(0) < 2K_\phi$. For this kind of mobile robots, using the relative error coordinates $\mathbf{e}_q = (e_x, e_y, e_\phi)$ we can see that if a Lyapunov function $V$ is the sum of positive definite functions of the errors, then the derivative of $V$ cannot be negative definite, but only a negative semidefinite function. This is a direct consequence of the nonholonomic constraint: if $(e_x, e_\phi)$ are null at a time, then $\dot{e}_y$ must be also null because of Eq. (3). Therefore $e_y$ cannot decrease at this moment, and hence neither does $V$. Therefore we cannot impose the derivative of $V$ to be negative definite, but only negative semidefinite. We designed the control law based on the following theorem.

**Theorem 2:** *For unicycle robots and using the projection given by* (7), $\mathbf{e}_q = (e_x, e_y, e_\phi) = 0$ *is an asymptotically stable equilibrium point, and* $\mathbf{u} \to \mathbf{u}_{\text{des}}(t) = \dot{r}\,\mathbf{u}_{\text{des}}(r)$, *if*:

- $\mathbf{u}_{\text{des}}(r) \neq 0$.
- *The desired path fulfills the uniqueness condition given by* (13).
- *The initial distance given by* (6) *satisfies* $d_q(0) < 2K_\phi$.
- *The following control is imposed*:
  1. $\mathbf{u}$ *is bounded and never null.*
  2. $$\begin{aligned}\big(e_x \cos(e_\phi) + e_y \operatorname{sen}(e_\phi)\big)\nu + (K_\phi e_\phi) K_\phi \omega \\ = -\Big[\tau_{xy}^{-1}\big(e_x \cos(e_\phi) + e_y \operatorname{sen}(e_\phi)\big)^2 \\ + \tau_\phi^{-1} K_\phi^2 e_\phi^2\Big] \end{aligned} \quad (14a)$$

     *or equivalently its limit when the projection* (7) *is satisfied and* $e_\phi \to 0$, $e_x \to 0$; *that is,*

$$\big(-K_\phi^2 \omega_{\text{des}} + \nu_{\text{des}} e_y\big)\nu + \big(K_\phi^2 \nu_{\text{des}}\big)\omega = 0 \quad (14b)$$

     *where* $\tau_{xy}^{-1}, \tau_\phi^{-1}, K_\phi > 0$.

*Proof:* Let us select the Lyapunov function (which matches with the semidistance):

$$V = \tfrac{1}{2} d_q^2(r,t) = \tfrac{1}{2}\big(e_x^2 + e_y^2 + K_\phi^2 e_\phi^2\big)$$

We must remark here that although we are using three error states in function $V$, the projection (7) is indeed considered in the control law. For example, if the projection were $e_x = 0$ (i.e., normal projection), the Lyapunov function would be $V = \tfrac{1}{2}(e_y^2 + K_\phi^2 e_\phi^2)$, as has been already proposed.[2,7] The peculiar thing here is that the projection (7) cannot be substituted directly on $V$, so we apply (7) in a different way. To be exact, control law 2 when $e_\phi \to 0$, $e_x \to 0$ has been derived using (7).

Differentiating and using state Eq. (3):

$$\dot{V} = \nu\big(e_x \cos(e_\phi) + e_y \operatorname{sen}(e_\phi)\big) + K_\phi^2 e_\phi \omega$$

Imposing control law 2, we find that this derivative $\dot{V}$ is negative semidefinite on errors $e_x, e_y$ and negative definite on error $e_\phi$:

$$\begin{aligned}\dot{V} &= \big(e_x \cos(e_\phi) + e_y \operatorname{sen}(e_\phi)\big)\nu + (K_\phi e_\phi) K_\phi \omega \\ &= -\Big[\tau_{xy}^{-1}\big(e_x \cos(e_\phi) + e_y \operatorname{sen}(e_\phi)\big)^2 + \tau_\phi^{-1} K_\phi^2 e_\phi^2\Big]\end{aligned}$$

Then $V$ (and $d_q$) is nonincreasing, so $\dot{V} \to 0$, and $V$ converges to some limit: $V \to V_{\lim} \geq 0$. Therefore by Barbalat's lemma[36] $e_\phi \to 0$, $\dot{e}_\phi \to 0$. At this limit, $\dot{V}$ is negative definite on error $e_x$, so $e_x \to 0$, $\dot{e}_x \to 0$, $e_y \to e_{y,\lim} < \infty$, and the control law is

$$\big(-K_\phi^2 \omega_{\text{des}} + \nu_{\text{des}} e_{y,\lim}\big)\nu + \big(K_\phi^2 \nu_{\text{des}}\big)\omega = 0$$

And state equations tend to

$$\begin{pmatrix} -\nu_{\text{des}}(t) + \omega_{\text{des}}(t) e_{y,\lim} + \nu \\ 0 \\ \omega - \omega_{\text{des}}(t) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Therefore by the third state equation $\omega \to \omega_{\text{des}}(t)$ and by the first state equation and control law

$$e_{y,\lim}\big(K_\phi^2 \omega_{\text{des}}^2 + \nu_{\text{des}}^2 - \nu_{\text{des}} \omega_{\text{des}} e_{y,\lim}\big) = 0$$

This gives two convergence solutions:

$$e_{y,\lim} = 0$$

$$\nu_{\text{des}} \omega_{\text{des}} e_{y,\lim} = K_\phi^2 \omega_{\text{des}}^2 + \nu_{\text{des}}^2$$

If $\nu_{\text{des}}$ or $\omega_{\text{des}}$ is null, only the first solution remains, i.e., $e_{y,\lim} = 0$. Let us define $K_{\text{des}} = \omega_{\text{des}}/\nu_{\text{des}}$ (assuming $\nu_{\text{des}}$ is not null), the planar curvature of the desired path. Then the second solution can be written as $e_{y,\lim} = K_\phi^2 K_{\text{des}} + 1/K_{\text{des}}$, for which the absolute minimum is $\min(e_{y,\lim}) = 2K_\phi$. Then this second solution will never be reached because $d_q$ is nonincreasing and provided that $d_q(0) < 2K_\phi$. Therefore $e_y(t) \to 0$ and due to the first state equation $\nu \to \nu_{\text{des}}(t)$. ■

**Remark 1:** $\nu_{\text{des}}(t)$ and $\omega_{\text{des}}(t)$ can be positive or negative depending on the sign of $\dot{r}$. As the convergence is implemented like a path following, the robot is intended to converge to a path, and the control algorithm should choose the required sign.

**Remark 2:** The theorem does not require extreme exigencies on input $\mathbf{u}$, except their circumscription to the line on the inputs plane given by 2. The proposed motion exigency (5) is good enough for SIRIUS, while other motion exigencies would be valid to ensure error convergence. Equations (14) represent a line in the normalized inputs plane $(\nu, K_\phi \omega)$. The intersection of motion exigency (5)

and line (14) will give us the requested values for $\nu$ and $\omega$ (for the allowed maximum errors of SIRIUS there are always two solutions). Moreover, the intersection of (5) and (14) reduces to a first-order differential equation when $e_\phi \to 0$ or $e_x \to 0$. In these cases, the parameters $\tau_{xy}, \tau_\phi$ play the role of *time constant*.

## 6. EXPERIMENTAL AND SIMULATION RESULTS

In this section we show the behavior of our system in several situations. For our system, the user and the electric wheelchair weight are both considerable. Hence inertia load driven by the motors is important even when the gearbox ratio is high (31:1 in SIRIUS). Despite this and due to the considerations explained before, the PF method ensures that errors will not grow too much when motor response is slow.

### Simulation Results

Even when asymptotic convergence is ensured, simulation is always a good way to verify and observe the control behavior. This behavior is primarily significant when errors are big, because if they are small the simulated system's behavior should be similar to that of an exponential convergence system. Besides, a *real* (2,0) robot like SIRIUS cannot do piecewise paths including planar curvature discontinuities without stopping (e.g., a straight line plus a circle). This is due to the motor's inertia, which prevents instantaneous changes of wheel speeds, i.e., from the instantaneous changes of linear or angular velocities that a curvature discontinuity requires. The problem of these discontinuities will be analyzed below.

In this subsection, we have selected two very interesting examples that comply with the curvature continuity: (1) approaching to a straight line, the most frequent convergence case found in the current literature, and (2) converging to a zero-radius turn, where our PF shows its generality. The values for constant parameters in these examples have been tuned to ensure a smooth convergence: $\tau_{xy} = 0.5$ s, and $\tau_\phi = 0.5$ s. The whole motion is $K_{\text{mov}} = 0.5$ m/s, and the constants $b_\phi = K_\phi$ are 0.25 m.

In the first case we have selected large initial errors to prove the good convergence of the method in the presence of extreme conditions (Fig. 6). The error $e_x$ must always be zero according to the projection (7) and the solid line for the error $e_y$ gives us

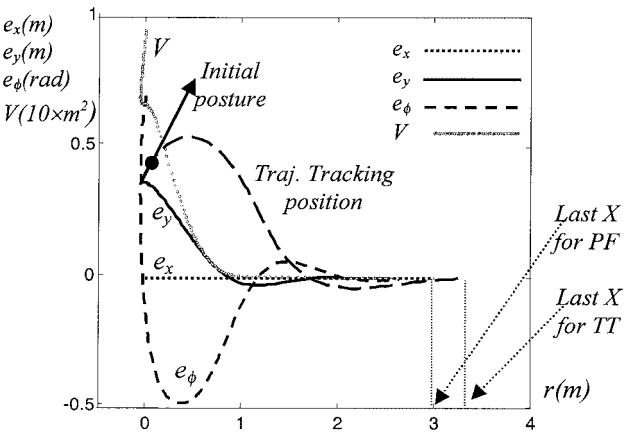


**Figure 6.** PF to a line ($X$ axis) under large initial errors.

the real robot trajectory. Note that the robot begins the following in reverse direction at the first transient to reduce the errors faster, and, thus, parameter $r$ decreases too.

In this example it is very interesting to compare PF with trajectory tracking. We have used for TT the control law of ref. 4 (its trajectory is the dashed line), tuning its constants so it behaves in a similar fashion for small errors ($K_x = 20\ \text{s}^{-1}$; $K_y = 6.0\ \text{cm}^{-1}$; $K_\theta = 3.2\ \text{cm}^{-1}$). Note that the last $X$ position for TT is almost 3.5 m (7 s $\times$ 0.5 m/s = 3.5 m), while the last $X$ for PF is less than 3 m. This is because in a pure TT where $r = t$ the reference robot "pulls" the real one and it will advance the same distance as the reference trajectory. In the end the TT method introduces more oscillations than PF, and it has a transient response that separates the robot from the desired line. This is a well-known advantage of PF.[7,11]

In the second example (Fig. 7) the desired trajectory is the vertical axis, because the reference robot turns around its point $P_o$ (see Fig. 1). The real robot movement would be given by the projection on plane $XY$. As in the previous case, although initial errors were big, PF chooses the nearest point on the desired trajectory, i.e., the point with $e_\phi = 0$ [according to projection (7)]. Note that in PF inputs are split into $\nu$ and $\omega$ in the most convenient form according to the control law.

### Experimental Evaluation

We use the feedback given by the two optical encoders at each driven wheel to collect and calculate real trajectories in SIRIUS. When the user is driving

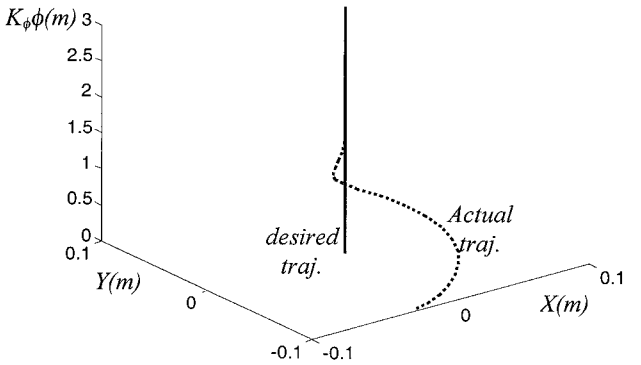


**Figure 7.** PF to a zero-radius turn ($e_x(0) = -0.03$ m, $e_y(0) = -0.1$ m).

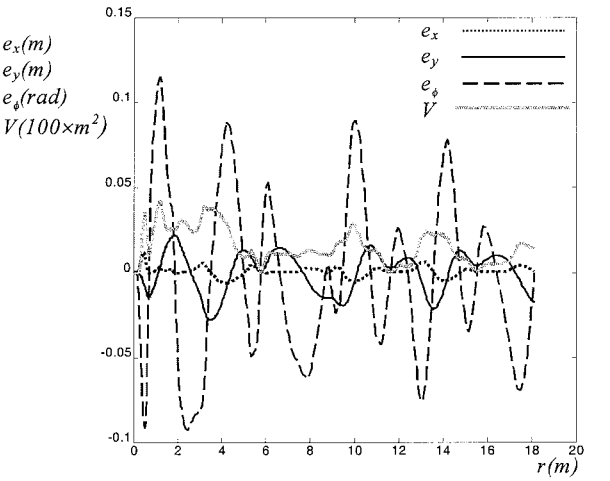


**Figure 8.** Errors recovering a typical trajectory in SIRIUS.

his or her joystick, the wheel positions are continuously memorized. To save memory only a pair of encoder values ($N_R$, $N_L$) are stored when the sum of their square increments exceeds a certain value $\Delta_{\max}$, namely,

$$(\Delta N_R)^2 + (\Delta N_L)^2 \geq \Delta_{\max}$$

In SIRIUS real trajectories are very smooth because system inertia is very large. So the error introduced by this storing method is negligible even when $\Delta_{\max}$ is large, while the memory saving is considerable. Also every time the chair is stopped, the point on the memorized path is specially labeled because it may mean a desired curvature discontinuity.

When the user presses the trajectory recovery button, the desired linear and angular velocities are calculated to obtain the input file contained in two arrays, where their index plays the role of the descriptor parameter $r$. At the first moment initial errors are obviously null. Then recovery begins and state equations are integrated via the second order Runge–Kutta method, considering the projection and the movement exigency. To depict the desired and recovered paths, velocities and errors are saved in a disk file at the same time that the recovery is completing. Afterwards desired and real positions ($X_{\text{des}}$, $Y_{\text{des}}$ and $X, Y$) are calculated off-line.

Experimental results show us that the lateral error $e_y$ is always under 3 cm. In Fig. 8 we show the errors for a typical trajectory recovery, where the three coordinates ($e_x$, $e_y$, $e_\phi$) are slightly oscillating mainly because of the motor's response delay (even though this effect has been reduced through the addition of an anticipatory control in the inner control loop). In Fig. 9 we compare the desired path in the $XOY$ plane trajectory with the real following.

The slippage accumulations introduce another deviation from the path that the user previously did. In SIRIUS this is minimized by doing the whole recovery at low speed ($K_{\text{mov}} = 0.4$ m/s) and bounding the maximum velocity when user is driving the chair. Approximately there are another 2 or 3 cm of lateral error for every 10 m. We should incorporate the feedback of another external sensor if we would contemplate and eliminate this kind of deviation,

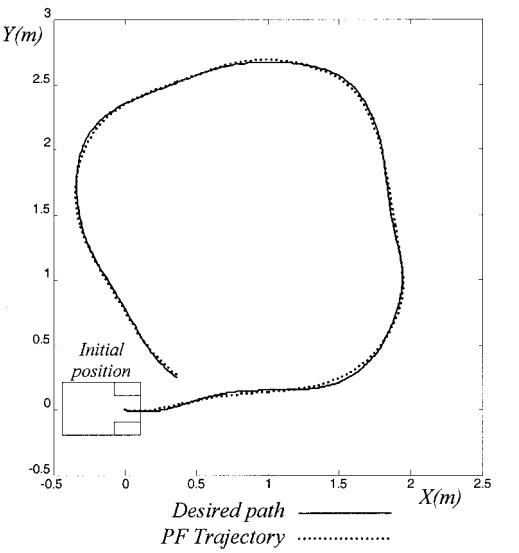


**Figure 9.** Real and desired paths recovering a typical trajectory in SIRIUS.

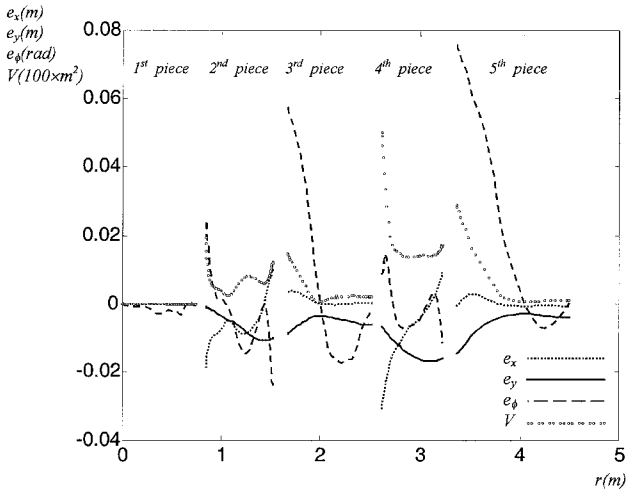


**Figure 10.** Errors recovering a piecewise trajectory in SIRIUS.

but in SIRIUS it is not usual to recover a longer path.

Another interesting case is when a desired trajectory contains singular points. In the next experiment (Figs. 10 and 11) the user has made a path composed of five pieces: the first, third, and fifth are almost straight lines and the other two are almost zero-radius left turns (that will be right turns in the reverse recovery). When the chair is approaching a singular point, $K_{\mathrm{mov}}$ is progressively decreased until SIRIUS is sufficiently near to it. Then the new

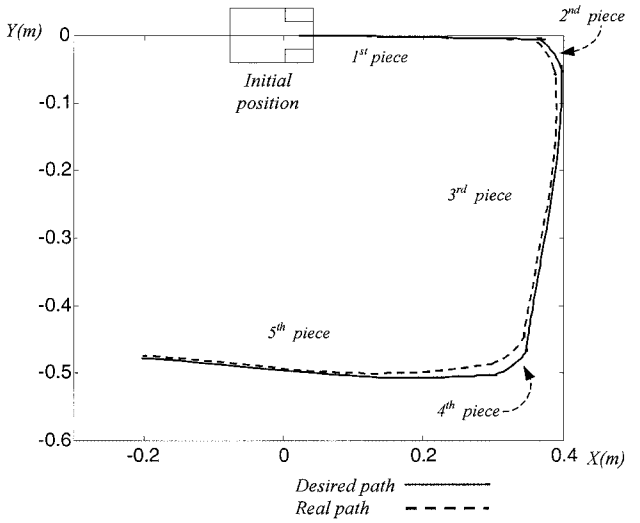


**Figure 11.** Real and desired paths recovering a piecewise trajectory in SIRIUS.

projecting point on the second piece is calculated, and the coordinate change from the first to the second piece is done (as shown in Fig. 5). This coordinate change introduces an error discontinuity that will be recovered by the control (again some little oscillation remains mainly due to the motor's response delay). So the discontinuity's magnitude depends largely on how near to the singular point the chair is. All of these facts in this change are empirically adjusted to ensure that the chair's velocity is sufficiently low and the error discontinuity is small. Note that a specific control to approach closer to the singular point is not necessary because this intermediate point is not the user's goal. The values for constant parameters in these examples are tuned to ensure a faster convergence in SIRIUS: $\tau_{xy} = 0.3$ s, $\tau_\phi = 0.3$ s.

## 7. CONCLUSIONS

We have presented a new path following for unicycle mobile robots and evaluated it in a computerized wheelchair that recovers the paths done by the user in reverse direction. The path following has been designed to be valid for all the possible trajectories. It relates the actual robot posture to the desired path via a geometric projection that considers all the error coordinates. In fact, it chooses the desired path's point whose distance to the robot is minimal. Therefore the projection can be applied also to desired paths such as straight lines, circles, or zero-radius turns. In the first case the projection will choose the desired point with null longitudinal error, and in the last case it will choose that with null orientation error. Hence the projection emphasizes linear velocity when following a line and angular velocity when approaching to a pure turn. In addition, the uniqueness of the projection is carefully analyzed. As a result, a slight bound is needed for the curvature derivative of desired paths (for bounded errors). Although we have to impose this bound to the collected paths done by the user with his or her joystick, this does not limit the wheelchair's maneuverability, because the desired paths that are out of the bound are very abrupt. Moreover we impose a ''motion exigency'' to force the robot to move. This exigency permits the total robot's movement to be inverted in angular or in linear speed. Finally an asymptotically stable control law is found using the closed form equation of the proposed path following and the second Lyapunov method. The evaluation of the path following

and the control law on the wheelchair shows that its behavior is robust under the high perturbations of this system and under high initial errors for any trajectory driven by the user.

The authors thank Professor Claude Samson (INRIA, La France) for his help and his kind mailing of research reports.

## REFERENCES


1. M. Bloch, M. Reyhanoglu, and N.H. McClamroch, Control and stabilization of nonholonomic dynamic systems, IEEE Trans Automat Contr 37 (1992), 1746–1757.
2. A. Micaelli and C. Samson, Trajectory tracking for unicycle-type and two-steering-wheels mobile robots, Institut National de Recherche en Informatique, et en Automatique, Rapport de Recherche 2097, 1993.
3. M. Sampei, T. Tamura, T. Itoh, and M. Nakamichi, Path tracking control of trailer-like mobile robot, Proc IEEE/RSJ Int Workshop Intelligent Robots and Systems, IROS'91, Osaka, 1991, pp. 193–198.
4. Y. Kanayama et al., A stable tracking control method for an autonomous mobile robot, Proc 1990 IEEE Int Conf Robotics and Automation, Cincinnati, 1990, pp. 384–389.
5. F. Lamiraux, S. Sekhavat, and J.P. Laumond, Motion planning and control for Hilare pulling a trailer, IEEE Trans Robotics and Automat 15(4) (1999).
6. D.-H. Kim and J.-H. Oh, Nonlinear tracking control of trailer systems using the Lyapunov direct method, J Robotic Syst 16 (1999), 1–8.
7. C. Canudas de Wit, H. Khennouf, C. Samson, and O.J. Sordalen, ''Nonlinear control design for mobile robots,'' Recent trends in mobile robots, Y.F. Zheng (Editor), World Scientific Series in Robotics and Automated Systems, Singapore, 1993.
8. F. Díaz del Río, A. Civit Balcells, G. Jiménez, and J.L. Sevillano, Path tracking in the SIRIUS wheelchair, The 4th European Conference for the Advancement of Technology, AAATE Conference 1997, Thessaloniki, Greece, IOS Press, Amsterdam, 1997
9. F. Díaz del Río, G. Jiménez, J.L. Sevillano, S. Vicente, and A. Civit Balcells, A generalization of path following for mobile robots, Proc 1999 IEEE Int Conf Robotics and Automation, ICRA'99, Detroit, May 1999.
10. R.M. DeSantis, Path-tracking for a tractor-trailer-like robot, Int J Robotics Res 13(6) (1994).
11. N. Sarkar, X. Yun, and V. Kumar, Control of mechanical systems with rolling constraints: application to dynamic control of mobile robots, Int J Robotic Res 13(1) (1994).
12. C. Samson, Path following and time-varying feedback stabilization of a wheeled mobile robot, Proc Int Conf, ICARCV'92, Singapore, 1992, p. RO-13.1.
13. C. Altafini, A path-tracking criterion for an LHD articulated vehicle, Int J Robotics Res 18 (1999), 435–441.
14. O.J. Sordalen and C. Canudas de Wit, Exponential control law for a mobile robot: extension to path following, IEEE Trans Robotics Automat 9(6) (1993).
15. P. Manfredo do Carmo, Differential geometry of curves and surfaces, Prentice-Hall, Englewood Cliffs, NJ, 1976.
16. G. Walsh, D. Tilbury, S. Sastry, R. Murray, and J.P. Laumond, Stabilization of trajectories for systems with nonholonomic constraints, IEEE Trans Automat Cont 39(1) (1996).
17. G. Oriolo, S. Panzieri, and G. Ulivi, An iterative learning controller for nonholonomic mobile robots, Int J Robotics Res 17 (1998), 954–970.
18. G. Oriolo, S. Panzieri, and G. Ulivi, Learning optimal trajectories for non-holonomic systems, Int J Cont 73(10) (2000).
19. R. Colbaugh, E. Barany, and K. Glass, Adaptive control of nonholonomic robotic systems, J Robotic Syst 15(7) (1998), 365–393.
20. C. Samson, Control of chained systems: application to path following and time-varying point-stabilization of mobile robots, IEEE Trans Automat Cont 40 (1995), 64–77.
21. O.J. Sørdalen and O. Egeland, Exponential stabilization of nonholonomic chained systems, IEEE Trans Automat Cont 40 (1995), pp. 35–49.
22. L.G. Busnell, D.M. Tilbury, and S.S. Sastry, Steering three-input nonholonomic systems: the fire truck example, Int J Robotic Res 14 (1995), 366–373.
23. A. Balluchi, L. Benvenuti, M. DiBenedetto, C. Pinello, and A. Sangiovanni-Vincentelli, Hybrid control in automotive applications, Proc IEEE 88(7) (2000), pp. 888–912.
24. B. Thuilot, B. d'Andrea-Novel, and A. Micaelli, Modeling and feedback control of mobile robots equipped with several steering wheels, IEEE Trans Robotics Automat 12(3) (1996).
25. A. De Luca, G. Oriolo, and C. Samson, ''Feedback control of a nonholonomic car-like robot,'' Robot motion plannning and control, J.-P. Laumond (Editor), Springer-Verlag, Berlin, 1998, pp. 171–253.
26. A. Civit Balcells, F. Díaz del Río, J.L. Sevillano, and G. Jiménez, SIRIUS: a low cost high performance computerized wheelchair, Proc Int Workshop Medical Robots, Vienna, 1996, pp. 23–30.
27. A. Civit Balcells, F. Díaz del Río, G. Jiménez, and J.L. Sevillano, ''A proposal for a low cost advanced wheelchair architecture,'' The 4th European Conference for the Advancement of Technology, AAATE Conference 1997, Thessaloniki, Greece, IOS Press, Amsterdam, 1997.
28. J. Frederiksen et al. ''Impairment, disability and handicap,'' Issues in telecommunication and disability, S. von Tezchner (Editor), Commission of the European Communities, Luxemburg, 1991.
29. G. Bourhis and P. Pino, Mobile robotics and mobility assistance for people with motor impairments: rational justification for the VAHM Project, IEEE Trans Rehabil Eng 4(1) (1996).
30. G. Campion, G. Bastin, and B. d'Andrea-Novel, Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, IEEE Trans Robotics Automat 12(1) (1996).
31. I.J. Cox, Blanche—an experiment in guidance and navigation of an autonomous robot vehicle, IEEE Trans Robotics Automat 7(2) (1991).

32. F. Díaz del Río, Analysis and evaluation of mobile robot control: application to electric wheelchairs (in Spanish). Ph.D. Thesis, University of Seville (Spain), 1997.
33. R.W. Brockett, ''Asymptotic stability and feedback stabilization,'' Differential geometric control theory, Birkhauser, Basel, 1983, pp. 181–208.
34. C.E. Thorpe (Editor), Vision and navigation: the Carnegie Mellon Navlab, Kluwer Academic Publishers, Norwall, MA, 1990.
35. R.S. Millman, Elements of differential geometry, Prentice Hall, Englewood Cliffs, NJ, 1977.
36. S. Sastry, Nonlinear systems: analysis, stability and control, Springer-Verlag, New York, 1999.