Path following using dynamic transverse feedback linearization for car-like robots

Adeel Akhtar  Christopher Nielsen  Steven L. Waslander

Abstract—This paper presents an approach for designing path following controllers for the kinematic model of car-like mobile robots using transverse feedback linearization with dynamic extension. This approach is applicable to a large class of paths and its effectiveness is experimentally demonstrated on a Chameleon R100 Ackermann steering robot. Transverse feedback linearization makes the desired path attractive and invariant while the dynamic extension allows the closed-loop system to achieve the desired motion along the path.

I. INTRODUCTION

The problem of generating accurate motion along a given path for a control system can be broadly classified as either a path following problem or a trajectory tracking problem [1]. In a path following problem, unlike trajectory tracking, the main task of the controller is to follow a path with no a priori time parameterization associated to the motion along the path.

The extra degree of freedom in path following of assigning the timing law associated with path traversal allows a significant improvement in the achievable performance for non-minimum phase systems [2], [1]. Another key advantage of adopting the path following approach is that the path can be made an invariant set for the closed-loop system. In the context of mobile robotics, this means that once the mobile robot is on the path, with appropriate orientation, it never leaves the path.

On the other hand, since a tracking controller tracks a specific system trajectory, if the robot is initialized on the path but its position does not coincide with the reference position the robot may leave the path before asymptotically approaching the reference point on the path again [3]. In this paper we design a path following controller for the kinematic model of a car-like robot [4]. This model approximates the mobility of a car and is relevant in automated driving applications. Moreover, the car-like robot is the simplest nonholonomic vehicle that displays the general characteristics and the difficult maneuverability of higher dimensional systems, e.g., of a car towing trailers [4]. Accurate movement along a path is desirable for car-like robots when they operate in tight spatial conditions, like indoor robots moving in a room with obstacles. In these cases, path following can be used in conjunction with path planning to achieve collision-free motion.

Trajectory tracking and internal stability of the car-like robot were analyzed in [5]. The performance of the controller therein was tested both in simulation and on an experimental testbed. Path following controllers were proposed for the car-like robot in [6], [7], [8]. The approach in [6] is similar to the one followed in this paper. The key difference is that we do not fix the translational velocity of the car and consequently the path can be rendered invariant while having variable dynamics along the path. In [9], [10] a similar problem is solved in the presence of phase constraints and limited control resources. The car-like robot is treated as a single input system, and the translational velocity is a given, sign-definite, possibly time-varying, function. In [11] it was shown that transverse feedback linearization can be used to design path following controllers for the car-like robot using only the steering input. In this paper we provide explicit expressions for feedback control laws that achieve path following while allowing the motion along the path to change.\(^1\)

Feedback linearization controllers are criticized because they only work “perfectly” in simulation, i.e., in the absence of disturbances and parameter uncertainty. The authors of [13] highlight that dynamically extended feedback linearized controllers can involve high-order derivative terms which can be sensitive to sensor noise and modeling uncertainty making them difficult to implement experimentally. Furthermore, the sensors used to estimate the states of car-like robots are relatively inaccurate with lower update rates. These practical constraints make the implementation of the proposed controller challenging. Experimental implementation of a path following controller using sliding mode control was presented in [8]. A reference tracking and set-point regulation dynamic feedback linearization controller was presented in [14].

A large class of non-linear systems fall in the category of differentially flat systems [15]. Finding a flat output is, in general, difficult and involves finding a function that satisfies the conditions given in [16]. The search for a flat output can be simplified by noting that they often have strong geometric interpretations [17]. In [18] a flatness based approach is used to derive open-loop control laws for a kinematic car-like robot that are combined with interpolation using \(G^2\)-splines. In this paper we choose a virtual output because it has very strong physical meaning for the path following problem and subsequently show that it is a flat output. We use dynamic extension [19] of the original system to achieve the desired relative degree of the closed-loop system. Every system which is feedback linearizable via dynamic extension is differentially flat [20]. While we consider a kinematic model, the proposed

\(^1\) A preliminary version of this paper, without Sections IV, V and VI, was presented in [12].
controller can be extended to dynamic models using integrator backstepping [21].

A. Contributions

The main contributions of this paper are 1) an approach to designing path following controllers for car-like vehicles that is physically intuitive and mathematically proven to achieve invariance of the path while traversing the path with desired dynamics. 2) A method for approximating arbitrary smooth parameterized paths as the zero level set of a function. 3) Experimental results that demonstrate accurate path following and feasibility of the proposed approach.

B. Notation

Let \( \text{col}(x_1, \ldots, x_n) = [x_1 \cdots x_n]^\top \in \mathbb{R}^n \) where \( ^\top \) denotes transpose. We denote the Euclidean inner product by \( (x, y) \) and the associated Euclidean norm by \( \|x\| \). We let \( I_n \) represent the \( n \times n \) identity matrix and \( 0_{m \times n} \) represent the \( m \times n \) zero matrix. Given a set \( A \subset \mathbb{R}^n \), the point-to-set distance to \( A \) is denoted \( \|\cdot\|_A \). Given a function \( f : A \to B \), we let \( \text{Im}(f) \) or \( f(A) \) denote its image. A continuous function \( \alpha : [0, \infty) \to [0, \infty) \) is said to belong to class-\( \mathcal{K}_\infty \) if \( \alpha(0) = 0 \) and it is strictly increasing [22]. Given a \( C^1 \) mapping \( \phi : \mathbb{R}^n \to \mathbb{R}^m \) let \( d\phi_x \) be its Jacobian evaluated at \( x \in \mathbb{R}^n \). If \( f, g : \mathbb{R}^n \to \mathbb{R}^n \) are smooth vector fields we use the following standard notation for iterated Lie derivatives of the form:

\[
L^0_f \phi := \phi, \quad L^1_f \phi := L_f(\dot{L}^{k-1}_f \phi) = (dL^{k-1}_f \phi_x, f(x)), \quad L_g L_f \phi := L_g(L_f \phi) = (dL_f \phi_x, g(x)).
\]

II. PROBLEM FORMULATION

Consider the kinematic model\(^2\) of a car-like robot with rear traction

\[
\dot{x} = \begin{bmatrix}
\cos x_3 & 0 \\
\sin x_3 & 0 \\
\frac{1}{2} \tan x_4 & 0 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
v \\
\omega
\end{bmatrix}
\]

where \( x \in \mathbb{R}^4 \) is the state, \( v \in \mathbb{R} \) is the translational speed and \( \omega \in \mathbb{R} \) is the angular velocity of the steering angle \( x_4 \). We impose a steering angle constraint

\[
(\forall t \geq 0) \quad -\frac{\pi}{2} < -\pi_4 \leq x_4(t) \leq \pi_4 < \frac{\pi}{2}
\]

where \( \pi_4 > 0 \) is given. We take the car’s position in the plane as the output of (1)

\[
y = h(x) = \text{col}(x_1, x_2).
\]

Suppose we are given a curve \( C \) in the output space \( \mathbb{R}^2 \) of (1) as a regular parameterized curve

\[
\sigma : \mathbb{D} \to \mathbb{R}^2, \quad \lambda \to \text{col}(\sigma_1(\lambda), \sigma_2(\lambda)),
\]

where \( \sigma \in C^r \) with \( r \geq 3 \) and \( C = \text{Im}(\sigma) \). Since \( \sigma \) is regular, without loss of generality, we assume it is unit-speed parameterized, i.e., \( \|\sigma(\cdot)\| \equiv 1 \). Under this assumption, the curve \( \sigma \) is parameterized by its arc length. For closed curves with finite length \( L \), this means that \( \mathbb{D} = \mathbb{R} \mod L \) and \( \sigma \) is \( L \)-periodic, i.e., for any \( \lambda \in \mathbb{D} \), \( \sigma(\lambda + L) = \sigma(\lambda) \). When the curve is not closed \( \mathbb{D} = \mathbb{R} \). We impose geometric restrictions on the class of curves considered [23].

Assumption 1 (submanifold). The curve \( C \) is a one-dimensional embedded submanifold of \( \mathbb{R}^2 \).

Assumption 1 imposes that the path has no self-intersections, no “corners”, and does not approach itself asymptotically.

Assumption 2 (implicit representation). The curve \( C \subset \mathbb{R}^2 \) has implicit representation \( C = \{ y \in W : s(y) = 0 \} \) where \( s : W \subset \mathbb{R}^2 \to \mathbb{R} \) is a smooth function such that \( ds_y \neq 0 \) on \( C \) and \( W \) is an open set. Moreover, there exist two class-\( \mathcal{K}_\infty \) functions \( \alpha, \beta : [0, \infty) \to [0, \infty) \) such that

\[
(\forall y \in W) \quad \alpha(\|y\|) \leq \|s(y)\| \leq \beta(\|y\|).
\]

Assumption 2 asks that the entire path be represented as the zero level set of the function \( s \). This is always possible, locally, if Assumption 1 holds. The second part of Assumption 2 ensures that, when \( C \) is not bounded, \( s(y) \to 0 \) if and only if \( y \to C \).

Since \( ds_x = I_2 \) for the output (3), the map \( h : \mathbb{R}^4 \to \mathbb{R}^2 \) is transversal [24] to \( C \) and therefore, if Assumption 1 holds, the lift of \( C \) to \( \mathbb{R}^4 \)

\[
\Gamma := (s \circ h)^{-1}(0) = \{ x \in \mathbb{R}^4 : s(h(x)) = 0 \}
\]

is a three dimensional submanifold.

Assumption 3 (curvature constraint [18], [25]). Given a steering angle constraint (2) the curvature \( \kappa(\lambda) \) of (4) satisfies

\[
(\forall \lambda \in \mathbb{D}) \quad \kappa(\lambda) < \frac{1}{\ell} \tan(\pi_4).
\]

Assumption 3 ensures that the path is feasible, in light of the steering angle constraint, for the car-like vehicle.

Problem 1: Given a curve \( C \) satisfying Assumptions 1, 2, and 3 find, if possible, a smooth control law for (1), (3) of the form

\[
\dot{c} = a(x, \zeta) + b(x, \zeta)u
\]

with \( \zeta \in \mathbb{R}^2, u = (u_1, u_2) \in \mathbb{R}^2 \) such that for some open set of initial conditions \( U \times V \subset \mathbb{R}^4 \times \mathbb{R}^4 \) with \( C \subset h(U) \)

PF1 The solution \( (x(t), \zeta(t)) \) of the closed-system (1), (8) exists for all \( t \geq 0 \) and \( \|h(x(t))\|_C \to 0 \) as \( t \to \infty \).

PF2 The curve \( C \) is output invariant independent of the desired motion along the path, i.e., if properly initialized, then \( \|h(x(t))\|_C = 0 \) for all \( t \geq 0 \).

PF3 The system asymptotically tracks a given motion profile \( \sigma(\lambda^{ref}(t)) \) where \( \lambda^{ref} : \mathbb{R} \to \mathbb{D} \) is smooth and \( \lambda^{ref}(t) \) is uniformly bounded away from zero.
III. DIFFERENTIALLY FLAT PATH FOLLOWING OUTPUTS

The path following manifold, denoted \( \Gamma^* \), associated with
the curve \( \mathcal{C} \) is the maximal controlled invariant subset of the
lift (6). Physically it consists of all those motions of the car-like robot (1)
for which the output signal (3) can be made to remain on the curve \( \mathcal{C} \)
by suitable choice of control signal [3].

The path following manifold is the key object that allows
one to treat the path following problem as a set stabilization
problem. If the path following manifold can be made attractive
and controlled invariant for the closed-loop system then PF1
and PF2 are satisfied.

When we apply the above definition to the car-like robot or,
more generally, to any drift-less system, it is immediate that
\( \Gamma^* = \Gamma \). This is because one can trivially make the entire set \( \Gamma \)
controlled invariant by setting \( v = 0 \). This characterization of
\( \Gamma^* \) cannot be used to solve Problem 1 because path invariance
is not achieved independently of the motion along the path.
Stabilizing \( \Gamma \) with \( v|_\Gamma = 0 \) ensures path invariance (PF2) but
fixes the motion along the path.

On the other hand, when \( v = v \neq 0 \) is a fixed constant,
the path following manifold can be characterized [11] using the
steering input \( \omega \). Physically, this means that the car like robot
can be made to follow \( \mathcal{C} \) solely using its steering input. The
main deficiency with the solution presented in [11] is that PF3
cannot be satisfied since \( v \) is fixed. To overcome this difficulty
a time-scaled transformation was applied in [9], [10], which
made it possible to ensure path invariance for variable speed
\( v(t) \).

To overcome this problem let \( v = v + \zeta_1 \), where \( \zeta_1 \) is the
first state of our dynamic controller and \( v \neq 0 \) is constant.
We take the simplest possible structure for the control law (8)
and let \( \zeta_2 = \zeta_2 \). In order to finish defining the control law we
let \( \zeta_2 = u_1 \) where \( u_1 \) is a new, auxiliary input. To simplify
notation, henceforth we do not distinguish between physical
states of the system \( (x_1, x_2, x_3, x_4) \) and states of the controller
\( (\zeta_1, \zeta_2) \). Let \( x_5 := \zeta_1, x_6 := \zeta_2 \). Therefore the system we
study has the form
\[
\dot{x} = f(x) + g_1(x)u_1 + g_2(x)u_2
\]
and the set \( \Gamma \) in (6) is embedded in the extended state space \( \mathbb{R}^6 \).
The dynamic extension allows us to enforce PF2
independently of the function \( \lambda \text{rel}(t) \) from PF3. Similar ideas
have been applied to a tower crane model in [26].

A. Feedback linearization

We treat the path following problem as a set stabilization
problem and we follow the general approach of [3], [23].
In order to satisfy PF1 and PF2 we first stabilize the path
following manifold \( \Gamma^* \). Once the path manifold has been
stabilized we use the remaining freedom in the control law
to impose desired dynamics on the path and satisfy PF3.

Let \( \mathcal{N}(\mathcal{C}) \subset \mathbb{R}^2 \) denote a neighbourhood of the curve \( \mathcal{C} \).
The neighbourhood \( \mathcal{N}(\mathcal{C}) \) has the property that if \( y \in \mathcal{N}(\mathcal{C}) \)
then there exists a unique \( y^* \in \mathcal{C} \) such that \( ||y||_C = ||y - y^*|| \).
This allows us to define the function
\[
\varpi : \mathcal{N}(\mathcal{C}) \to \mathbb{D} \\
y \mapsto \arg \inf_{\lambda \in \mathbb{D}} ||y - \sigma(\lambda)||.
\]

This function is as smooth as \( \sigma \) is which, by assumption, is
at least \( C^3 \). Using (10) define the “path following output”
\[
\dot{y} = \begin{bmatrix} \frac{\pi(x)}{\alpha(x)} \\ \frac{\varpi \circ h(x)}{s \circ h(x)} \end{bmatrix}
\]

Let \( \Gamma_+ := \Gamma \cap \{ x \in \mathbb{R}^6 : x_5 + v > 0 \}, \Gamma_- := \Gamma \cap \{ x \in \mathbb{R}^6 : x_5 + v < 0 \} \). The next lemma shows that the output (11) yields a well-defined relative degree on \( \Gamma_+ \bigcup \Gamma_- \)
where \( \bigcup \) denotes disjoint union.

Lemma III.1. The dynamic extension of the car-like robot (9)
with output (11) yields a well-defined vector relative degree
of \( \{3, 3\} \) at each point on \( \Gamma_+ \bigcup \Gamma_- \).

Proof. Let \( x^* \in \Gamma_+ \bigcup \Gamma_- \) be arbitrary. By definition of \( \Gamma \)
the output \( h(x^*) \) is on the path \( \mathcal{C} \). Let \( \lambda^* \in \mathbb{D} \) be such that
\( h(x^*) = \sigma(\lambda^*) \). By the definition of vector relative degree we
must show that \( L_{g_1} L_{f} p_1(x) = L_{g_2} L_{f} p_2(x) = L_{g_3} L_{f} p_3(x) = 0 \)
for \( i \in \{0, 1\} \) in a neighbourhood of \( x^* \) and
that the decoupling matrix

\[
D(x) = \begin{bmatrix} L_{g_1} L_{f} p_1(x) \\ L_{g_2} L_{f} p_2(x) \\ L_{g_3} L_{f} p_3(x) \end{bmatrix}
\]

is non-singular at \( x = x^* \). Since
\[
\frac{\partial \sigma(x)}{\partial x_i} = \frac{\partial \sigma(x)}{\partial x_i} \equiv 0
\]
for \( i \in \{3, 4, 5, 6\} \), it is easy to check that \( L_{g_1} L_{f} p_1(x) = \)
\( L_{g_2} L_{f} p_2(x) = L_{g_3} L_{f} p_3(x) = 0 \) for \( i \in \{0, 1\}, j \in \{1, 2\} \).

To show that the decoupling matrix (12) is non-singular at
\( x = x^* \), we first find that
\[
\text{det}(D(x)) = \frac{(v + x_5)^2}{\ell \cos^2 x_4} (\sigma_1(\lambda^*) \partial_{x_2} \alpha - \sigma_2(\lambda^*) \partial_{x_3} \alpha).
\]

The only way for this determinant to vanish is if either (i) \( v = -x_5 \) or (ii) \( \sigma_1(\lambda^*) \partial_{x_2} \alpha - \sigma_2(\lambda^*) \partial_{x_3} \alpha = 0 \). Condition (i) does
not occur for \( x \in \Gamma_+ \bigcup \Gamma_- \). We now argue that condition (ii)
ever occurs on the path because the vectors \( \text{col}(\partial_{x_2} \alpha, \partial_{x_3} \alpha) \)
and \( \sigma'(\lambda^*) \) are orthogonal.

The chain rule and the form of the output map (3) yield
\[
\text{col}(\partial_{x_2}(\alpha(x)), \partial_{x_3}(\alpha(x))) = ds_{h(x^*)} = \begin{bmatrix} 0 \end{bmatrix}. \] By Assumption 2 the differential \( ds_{y} \neq 0 \) for \( y \in \mathbb{C} \). Thus the vector \( ds_{h(x^*)} \)
is a non-zero gradient vector and is orthogonal to the path
at \( h(x^*) \). On the other hand the vector \( \sigma'\lambda^* \) is non-zero
because \( \sigma \) is regular and also tangent to the curve. Hence
\[
\text{col}(ds_{h(x^*)}^T, \sigma'(\lambda^*)) = 0 \] and then the rotated vector \( \sigma'\lambda^* \)
is linearly dependent. Let \( R_\pi \) be a rotation of the plane by \( \pi/2 \). Then
\[
R_\pi ds_{h(x^*)} = k(\sigma(\lambda^*))\sigma'(\lambda^*)
\]
for some smooth, scalar-valued, non-zero function $k : \mathbb{R}^2 \to \mathbb{R}$. The function $k$ is never equal to zero because the vector $\text{d}h_x^T(\dot{x}_*)$ is never zero.

Returning to the expression for $\det(D(x))$, we have that

$$
\sigma_1'(\lambda^*) \partial_{x_1} \alpha - \sigma_2'(\lambda^*) \partial_{x_2} \alpha = \left< R_{x^*}^T h_x^T(\dot{x}_*)^T, \sigma'(\lambda^*) \right>
= k(\alpha(\lambda^*)) \langle \sigma'(\lambda^*), \sigma'(\lambda^*) \rangle
= k(\alpha(\lambda^*)) \| \sigma'(\lambda^*) \|^2
= k(\alpha(\lambda^*)).
$$

Let $\Gamma^* := \{ x \in \mathbb{R}^6 : \alpha(x) = L_f \alpha(x) = L_f^2 \alpha(x) = 0 \}$. Define $\Gamma^*_* := \Gamma^* \cap \Gamma_+$ and $\Gamma^*_* := \Gamma^* \cap \Gamma_-$. The next result defines a diffeomorphism valid in a neighbourhood of $\Gamma^*_*$. The equivalent result for $\Gamma^*_{-}$ is omitted to avoid repetition.

**Corollary III.2.** Let $x^* \in \Gamma^*_*$. There exists a neighbourhood $U_+ \subset \mathbb{R}^6$ containing $\Gamma^*_+$ such that $T : U_+ \to \Gamma^*_*+$(\eta_1 \eta_2 \eta_3 \xi_1 \xi_2 \xi_3) = T(x) = \begin{bmatrix} \pi(x) \\ L_f^2 \pi(x) \\ L_f^2 \pi(x) \\ L_f^2 \alpha(x) \\ L_f^2 \alpha(x) \end{bmatrix}
$$

is a diffeomorphism onto its image.

**Proof.** In order to show that (13) is a diffeomorphism in a neighbourhood of $\Gamma^*_*$, we appeal to the generalized inverse function theorem [24, pg. 56]. We must show that 1) for all $x \in \Gamma^*_*$, $dT_x : \mathbb{R}^6 \to \mathbb{R}^6$ is an isomorphism, and 2) $T|_{\Gamma^*_*} : \Gamma^*_* \to T(\Gamma^*_+)$ is a diffeomorphism. An immediate consequence of Lemma III.1 and [19, Lemma 5.2.1] is that the first condition holds.

To show that the second condition holds we explicitly construct the inverse of $T$ restricted to $\Gamma^*_*$. On $\Gamma^*_*$, $\xi_1(x) = \xi_2(x) = \xi_3(x) = 0$ and simple calculations show that the inverse of $T$ restricted to $\Gamma^*_*$ is

$$
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6 
\end{bmatrix}
= T|_{\Gamma^*_*}^{-1}(\eta, 0) = \begin{bmatrix}
\sigma_1(\eta_1) \\
\sigma_2(\eta_1) \\
\phi(\eta_1) \\
\arctan(\kappa(\eta_1)) \\
\eta_2 - \nu \\
\eta_3 
\end{bmatrix}
$$

where $\phi : \mathbb{D} \to \mathbb{R}$ is the map that associates to each $\eta_1 \in \mathbb{D}$ the angle of the tangent vector $\sigma'(\eta_1)$ to $C$ at $\sigma(\eta_1)$ and $\kappa : \mathbb{D} \to \mathbb{R}$ is the signed curvature. The inverse is clearly smooth which shows that $T|_{\Gamma^*_*}$ is a diffeomorphism onto its image.

This coordinate transformation of Corollary III.2 is physically meaningful for path following applications. When $\xi = 0$ the system is restricted to evolve on the path following manifold $\Gamma^*$. We call the $\xi$-subsystem the transversal subsystem and the states $\xi$ the transversal states. On the path following manifold the motion of the car-like robot on the path is governed by the $\eta$-dynamics. We call the $\eta$-subsystem the tangential subsystem and states $\eta$ the tangential states. When the robot is on the path following manifold, i.e., $\xi = 0$ then $\eta_1$ determines the position of the robot on the path, $\eta_2$ represents velocity of the robot along the path and $\eta_3$ represents acceleration of the robot along the path.

We apply the regular feedback transformation

$$
\begin{bmatrix}
u_1 \\
u_2 
\end{bmatrix}
:= D^{-1}(x) \begin{bmatrix}
- L_f \pi \\
- L_f^2 \alpha \n\end{bmatrix} + \begin{bmatrix} v^\parallel \\
v^\parallel \n\end{bmatrix}
$$

(14)

where $(v^\parallel, v^\parallel)$ are auxiliary control inputs. By Lemma III.1 this controller is well-defined in the neighbourhood of $\Gamma^*_*$ from Corollary III.2. Thus in a neighbourhood of $\Gamma^*_*$ the closed-loop system becomes

$$
\dot{\eta}_1 = \eta_2 \\
\dot{\eta}_2 = \eta_3 \\
\dot{\eta}_3 = v^\parallel \\
\dot{\xi}_1 = \xi_2 \\
\dot{\xi}_2 = \xi_3
$$

(15)

We refer to the control input $v^\parallel$ as the transversal input and $v^\parallel$ as the tangential input. The control law (14) has decoupled the transversal and tangential subsystems which makes designing $(v^\parallel, v^\parallel)$ to solve Problem 1 particularly easy. In summary, we have shown that the extended car-like robot is differentially equivalent to a controllable linear time invariant system (LTI) in a neighbourhood of each connected component, $\Gamma^*_*$ and $\Gamma^*_{-}$, of the path following manifold. Another way to state this is to say that the output (11) is a flat output for the car-like robot (1) [27, 28].

**B. Transversal and tangential control design**

The objective of the transversal controller is to force the system to converge to the path. For that we to stabilize the origin of the transversal subsystem. The simplest choice for the transversal input is

$$
v^\parallel(\xi) = k_1 \xi_1 + k_2 \xi_2 + k_3 \xi_3,
$$

with $k_i < 0$, $i \in \{1, 2, 3\}$, chosen so that the polynomial $s^3 - k_3 s^2 - k_2 s - k_1$ is Hurwitz. This controller exponentially stabilizes $\xi = 0$ and hence, under (5) in Assumption 2, makes the path following manifold attractive. These gains can be chosen using, for instance, pole-placement or quadratic optimization (LQR).

**Assumption 4** (desired motion on path), The desired motion on $C$ given by a smooth function $\eta^\text{ref}(t) := (\dot{x}^\text{ref}(t), \dot{\lambda}^\text{ref}(t), \lambda^\text{ref}(t))$, $t \geq 0$ with $| \dot{\lambda}^\text{ref} |$ uniformly bounded away from zero.

Assumption 4 ensures that $x_5 + \nu \neq 0$ for the desired motion. When $x_5 + \nu = 0$ the robot has no transational velocity, the decoupling matrix loses rank and the control law (14) is not well-defined. Given a desired motion that satisfies Assumption 4, let

$$
\begin{align*}
v^\parallel(\eta) &= k_4 (\eta_1 - \eta_1^\text{ref}(t)) + k_5 (\eta_2 - \eta_2^\text{ref}(t)) \\
&\quad + k_6 (\eta_3 - \eta_3^\text{ref}(t)) + \eta_1^\text{ref}(t),
\end{align*}
$$

(17)
with gains $k_i < 0$, $i \in \{4, 5, 6\}$, chosen so that the polynomial $s^3 - k_6s^2 - k_5s - k_4$ is Hurwitz. The numerical value of these gains can be chosen similarly to the transversal gains. Typically, one seeks that the closed-loop transversal dynamics converge to zero faster than the closed-loop tangential dynamics.

**Proposition III.3. The control law** (14), (16), (17) **solves Problem 1.**

**Proof.** Assume that $\hat{\eta}^{ref}_1 > 0$ and let $x(0) \in U_+$ where $U_+ \subseteq \mathbb{R}^k$ is defined in Corollary III.2. By Lemma III.1, and by shrinking $U_+$ if necessary, the control law (14), (16), (17) is well defined in $U_+$. The transversal controller (16) exponentially stabilizes $\xi = 0$ and hence, by Assumption 2, $x \to \Gamma^+_+ \subseteq U_+$ and PF1 holds. Since $\xi = 0$ is an equilibrium of the closed-loop transversal subsystem, if $x(0) \in \Gamma^+_+ \subseteq U_+$, $h(x(t))$ remains on the path for all future time. Therefore (16) achieves PF1, PF2 in Problem 1.

Define errors coordinates $e_\eta := \eta - \eta^{ref}$. It is straightforward to show that, for the tangential controller (17), that $e_\eta \to 0$. Once again, by shrinking $U_+$ if necessary, it is possible to ensure that $\eta_2(t) \neq 0$ during the transient phase in which $\eta_2 \to \eta_1$. This shows that the closed-loop tangential dynamics satisfy PF3 and hence Problem 1 is solved.

**Remark III.4.** When the desired motion corresponds to velocity tracking then $\eta^{ref}_1(0) = (0, \eta^{ref}_2(0), \eta^{ref}_3(t))$. In this case we select $k_1 = 0$. Similar comments apply to acceleration tracking.

**Remark III.5.** Proposition III.3 shows that the region of attraction\(^5\) is an open subset of $U_+$ [resp. $U_-$]. Since (13) relies (10), the image of this set under (3) must be a neighbourhood of $C$ in which the closest point on the path is well-defined. This is a necessary, far from sufficient, property of the region of attraction.

**IV. CURVE REPRESENTATION**

The control design technique discussed in this paper relies on having both a parametric representation (Equation (4)) and an implicit representation (Assumption 2) of the path $C$. Although such curve representation pairs are well known for many commonly used paths such as circles and Cassini ovals, not all cases can be addressed this way.

Given an arbitrary curve $C$ in $\mathbb{R}^2$ with a $C^r$, $r \geq 3$, regular parameterization (4), we provide the following procedure for finding its implicit representation. First, we approximate the parametric representation as a rational parametric curve using the Weierstrass approximation theorem. Second, relying on elimination theory [30], we represent the image of the rational approximation as an implicit function.

**A. Polynomial approximation**

Given $\sigma(\lambda) = \text{col}(\sigma_1(\lambda), \sigma_2(\lambda))$ we generate polynomial approximations $p_j(\lambda)$ to the functions $\sigma_j : \mathbb{D} \to \mathbb{R}$, $j \in \{1, 2\}$. We start by sampling the domain $\mathbb{D}$. Let $\{\lambda_1, \lambda_2, \cdots, \lambda_{q+1}\}$ be $q + 1$ points in $\mathbb{D}$ with $\lambda_i < \lambda_{i+1}$, $i \in \{1, \ldots, q\}$. Let $I := [\lambda_1, \lambda_{q+1}] \subseteq \mathbb{R}$. The associated points on $C$ are given by $\sigma(\lambda_i) = \text{col}(\sigma_1(\lambda_i), \sigma_2(\lambda_i))$, $i \in \{1, \ldots, q + 1\}$. If we seek a single polynomial $p_j(\lambda)$ of fixed order $N$ that approximates $\sigma_j(\lambda)$ at the sample points, then we can simply solve two least squares optimization problems for $p_j(\lambda) = \sum_{i=0}^{N} a^{j}_i \lambda^i$, $j \in \{1, 2\}$ to find the coefficients $a^{j}_i \in \mathbb{R}$. The Weierstrass Approximation Theorem [31] ensures that for any $\epsilon > 0$, there exists $N$ sufficiently large such that

$$\max_{\lambda \in I} \|\sigma_j(\lambda) - p_j(\lambda)\| < \epsilon. \quad (18)$$

Furthermore, at the sample points $\sigma(\lambda_i)$, the above polynomial approximation is optimal in the least squares sense.

A drawback of the above approach is that, for a given set $\{\lambda_1, \lambda_2, \cdots, \lambda_{q+1}\}$ and a given $\epsilon > 0$, the order $N$ of the polynomial required to ensure (18) holds is a priori unknown. In such cases we propose an algorithm to recursively compute polynomials $p_j$ that satisfy (18).

Our algorithm is based on the constructive proof of Weierstrass Approximation Theorem presented in [31]. In that proof, given a uniformly continuous function $f : [0, 1] \to \mathbb{R}$, one constructs a Bernstein polynomial

$$B_n^f(\theta) := \sum_{k=0}^{n} c_k h_n^k(\theta), \quad (19)$$

where $c_k := f(k/n)$ and $h_n^k(\theta) := \binom{n}{k} \theta^k (1 - \theta)^{n-k}$, which is shown to converge uniformly to $f$ as the order of $B_n^f$ gets sufficiently large. In our application, on the interval $I = [\lambda_1, \lambda_{q+1}]$, define the function $\tau : [0, 1] \to I$ as

$$\tau(\theta) = \lambda_1 + \theta (\lambda_{q+1} - \lambda_1).$$

This function is a homeomorphism between $[0, 1]$ and $I$. We use it to define

$$f_j(\theta) := \sigma_j \circ \tau(\theta), \quad j \in \{1, 2\}. \quad (20)$$

Using the Weierstrass Approximation Theorem and Bernstein polynomials for the function (20), we have that for any $\epsilon > 0$, there exists an integer $N > 0$ such that

$$(\forall n \geq N) \max_{\theta \in [0, 1]} \|f_j(\theta) - B_n^f(\theta)\| < \epsilon. \quad (21)$$

These arguments lead to the following conclusion.

**Lemma IV.1.** There exists a positive finite integer $N$ such that (18) holds with $p_j(\lambda) = B_n^f(\tau^{-1}(\lambda))$, $n \geq N$, $j \in \{1, 2\}$.

Algorithm 1, given below, shows how to use Lemma IV.1 to find the polynomials $p_j$.

**B. Sylvester matrix elimination method**

We now apply elimination theory to form an implicit representation of the planar curves obtained in Section IV-A. There are multiple ways to accomplish this but we present Sylvester’s method [32].
input : $\sigma_j : \mathbb{D} \rightarrow \mathbb{R}$
\[ \epsilon > 0 \]
\[ N = 1; \text{(start with the smallest possible order)} \]
\[ I = [\lambda_1, \lambda_{q+1}] \]
output: $p_j(\lambda)$
compute: $f_j = \sigma_j \circ \tau(\theta)$
while $error > \epsilon$ do
  for $k = 0; N$ do
    compute: $c_k = f_j(k/N)$
    compute: $h_N(\theta) = \binom{N}{k} \theta^k (1 - \theta)^{N-k}$
    compute: $p_j(\lambda) = p_j(\lambda) + c_k h_N(\tau^{-1}(\lambda))$
  end
  calculate error: $\max_{\lambda \in I} \| \sigma_j(\lambda) - p_j(\lambda) \|$
  $N = N + 1$
end
Algorithm 1: Curve approximation

Sylvester’s dialectic expansion [32] computes the resultant of a given polynomial system by constructing a matrix that is rank deficient whenever the polynomial system has a solution. Consider two polynomials constructed using the ideas of Section IV-A $p_1(\lambda) = \sum_{i=0}^n a_i \lambda^i$, $p_2(\lambda) = \sum_{i=1}^m b_i \lambda^i$. Let $y_1 = p_1(\lambda)$ and $y_2 = p_2(\lambda)$ and rewrite the polynomials as
\[ P_1(\lambda) := a_n \lambda^N + \cdots + a_1 \lambda + (a_0 - y_1) = 0 \]
\[ P_2(\lambda) := b_m \lambda^N + \cdots + b_1 \lambda + (b_0 - y_2) = 0. \]

The key insight in [33] is that, by viewing the terms $(a_0 - y_1)$ and $(b_0 - y_2)$ in (22) as constant terms, the associated resultant expresses the relationship which must exists among the coefficients in order for there to exist $\lambda$ that simultaneously satisfies both equations (22). In other words, the resultant itself is the implicit form of the parametric curve.

Let $\text{Syl} (P_1, P_2, \lambda)$ denote the $(n+m) \times (n+m)$ Sylvester matrix of $P_1$ and $P_2$ with respect to $\lambda$. Then, the resultant of $P_1$ and $P_2$ with respect to $\lambda$ is denoted by $\text{Res} (P_1, P_2, \lambda)$ and is given by $\text{Res} (P_1, P_2, \lambda) = \det (\text{Syl} (P_1, P_2, \lambda))$.

In summary, we use the following two step approach to implicitize a curve $C \subseteq \mathbb{R}^2$ with regular parameterization (4).

1) Approximate the function $\sigma_j : \mathbb{D} \rightarrow \mathbb{R}$, $j \in \{1, 2\}$ as a univariate polynomial $p_j(\lambda)$ over a compact interval. Do this using least squares optimization or Algorithm 1.

2) Form the Sylvester matrix $\text{Syl} (P_1, P_2, \lambda)$ using (22) then set $s(y) = \text{Res} (P_1, P_2, \lambda) = \det (\text{Syl} (P_1, P_2, \lambda))$ as the zero-level set representation of the approximation of $C$ obtained above.

V. IMPLEMENTATION ISSUES

In order to implement the proposed controller the coordinate transformation (13), the feedback (14) with $D(x)$ defined in (12) and transversal and tangential controllers (16), (17) must be computed. In this section we address the two main issues that complicate implementing the above. The first is that the parameterization of $C$ may not be unit-speed. The second is that the computation of the tangential states ($\eta_1$, $\eta_2$, $\eta_3$) involves computing the projection (10) and its derivatives. In general the function (10) does not have a closed-form which makes these calculations non-obvious.

Following the discussion in Section IV, we assume that we are given $C$, a not-necessarily unit-speed parameterization $\tilde{\sigma} : \mathbb{R} \rightarrow \mathbb{R}^2$, and a function $s : \mathbb{R}^2 \rightarrow \mathbb{R}$ such that Assumption 2 holds. Note that having an expression for $s(y)$ makes the computation of the transversal states straightforward using symbolic algebra software.

Let $\mathcal{N}(C) \subseteq \mathbb{R}^2$ be a neighbourhood of $C$. Now introduce a projection operator, defined in $\mathcal{N}(C)$, which is the same as (10) except it uses a non-unit speed parameterization
\[ \lambda^* = \tilde{\sigma}^*(y) = \arg \inf_{\lambda \in \mathbb{R}} \| y - \tilde{\sigma}(\lambda) \|. \] (23)

The value $\lambda^*$ can be effectively numerically computed using line search algorithms. For closed-curves this calculation is straightforward because the line search is over a compact interval of $\mathbb{R}$. For non-closed curves heuristic methods must be employed to compute the infimum. To calculate the first tangential state we find the arc-length
\[ \eta_1 = g(\lambda^*) := \int_0^{\lambda^*} \left\| \frac{d\tilde{\sigma}}{d\lambda} \right\| d\lambda \] (24)
so that $\eta_1 = g \circ \tilde{\sigma} \circ h(x)$. To calculate $\eta_2$ we note
\[ \eta_2 = \frac{\partial (g \circ \tilde{\sigma} \circ h)}{\partial x} dx = \left( \frac{\partial g}{\partial x} \right) \left. \frac{\partial \tilde{\sigma}}{\partial y} \right|_{y = h(x)} \left[ (v + x_5) \cos (x_3) \right] \left[ (v + x_5) \sin (x_3) \right]. \]

Simple geometric arguments, similar to those used in the proof of Lemma III.1, show that $\frac{\partial \tilde{\sigma}}{\partial y}$ is given by
\[ \frac{\partial \tilde{\sigma}}{\partial y} = \frac{(\tilde{\sigma}'(\lambda^*))^T}{\| \tilde{\sigma}'(\lambda^*) \|^2}. \] (25)
Differentiating (24) one obtains
\[ \frac{\partial g}{\partial \lambda} \bigg|_{\lambda = \lambda^*} = \| \tilde{\sigma}'(\lambda^*) \| \] (26)
and so
\[ \eta_2 = \frac{(\tilde{\sigma}'(\lambda^*))^T}{\| \tilde{\sigma}'(\lambda^*) \|^2} \left[ (v + x_5) \cos (x_3) \right] \left[ (v + x_5) \sin (x_3) \right]. \] (27)
To simplify notation let
\[ \Delta(x) := \frac{(\tilde{\sigma}'(\lambda^*))^T}{\| \tilde{\sigma}'(\lambda^*) \|^2}, \quad \Omega := \left[ \begin{array}{c} (v + x_5) \cos (x_3) \\ (v + x_5) \sin (x_3) \end{array} \right]. \]
To find $\eta_3$ we differentiate (27) and get $\eta_3 = \Delta \Omega + \Delta \Omega \dot{\Omega}$. The term $\Omega$ is easy to compute using the system dynamics (9). The term $\Delta = \Delta(\dot{\lambda})$ can be found by noting that
\[ \Delta := \frac{\partial \Delta}{\partial \lambda} = \frac{(\tilde{\sigma}'(\lambda^*))^T}{\| \tilde{\sigma}'(\lambda^*) \|^2} \left[ \| \tilde{\sigma}'(\lambda^*) \|^2 - (\tilde{\sigma}')^T \sum_{i=1}^2 \tilde{\sigma}'_{i} \tilde{\sigma}_{i} \right] \] (28)
and, using (24) and the chain rule,
\[ \dot{\lambda} = \frac{1}{\| \tilde{\sigma}'(\lambda^*) \|^2} \eta_2. \] (29)
This shows that the tangential state $\eta_3$ can be computed effectively using (25), (28), (29), $\Omega$ and $\dot{\Omega}$.

Finally, in order to implement the feedback transformation (14) we must find expressions for $L_7^2 \pi$ and the first
row of the decoupling matrix (12). The decoupling matrix is straightforward but tedious to compute, and calculations give

\[ L^3 \pi = \frac{\eta_2}{\|\sigma\|} \left( \Delta'\hat{\Omega} + \Delta'\Omega + \frac{d\Delta'}{dt} \Omega + \frac{\eta_2 \sum_{i=1}^{2} \sigma_i'\sigma_i}{\|\sigma\|^2} \Delta'\Omega \right) q + \Delta\hat{\Omega} + \Delta\Omega \]

(30)

where

\[ \Omega_2 := \frac{(v + x_5)}{\ell} \tan(x_4) \left[ \begin{array}{c} (1 - 2x_6) \sin(x_3) \\ (1 + 2x_6) \cos(x_3) \end{array} \right] \]

\[ - \frac{(v + x_5)^3}{\ell^2} \tan^2(x_4) \left[ \begin{array}{c} \cos(x_3) \\ \sin(x_3) \end{array} \right] . \]

Implementation of controller and the regular feedback (14) is summarized by Algorithm 2.

\textbf{Algorithm 2:} Control algorithm

\begin{itemize}
  \item \textbf{input}: \( \bar{\sigma}(\lambda) : \mathbb{R} \rightarrow \mathbb{R}^2 \) (non-unit speed)
  \item \textbf{system model} (9)
  \item \textbf{Current state} \( x \in \mathbb{R}^6 \)
  \item \textbf{output}: \( (u_1, u_2) \)
\end{itemize}

\begin{itemize}
  \begin{itemize}
    \item For each \( \lambda \) using \( (23) \) numerically compute \( \lambda^* \).
    \item Compute \( \bar{\sigma}'(\lambda^*), \bar{\sigma}''(\lambda^*), \bar{\sigma}'''(\lambda^*), ||\bar{\sigma}'(\lambda^*)|| \).
    \item Numerically compute \( \eta_1 \) using (24).
    \item Compute \( \eta_2 \) using (27).
    \item Compute \( \eta_3 \) using (25), (28), (29), \( \Omega \) and \( \hat{\Omega} \).
    \item Compute \( L^3_3 \pi \) using expression (30).
    \item Compute \( \xi_1, \xi_2, \xi_3, L^3_3 \alpha \).
    \item Compute \( (u_1, u_2) \) using (14), (16), (17).
  \end{itemize}
\end{itemize}

VI. EXPERIMENTAL VERIFICATION

A. Experimental platform and setup

The Chameleon R100 robot is a low cost car-like robot for testing control and estimation algorithms. A DC motor is attached to the rear axle of the robot. A servo motor is used to control the steering angle of the robot. The maximum steering angle is approximately \( \pi_4 = 0.4712 \) rad (27 degrees). This means that Assumption 3 is satisfied, given that \( \ell = 22.9 \) cm in this case, if the maximum curvature of the path is 2.22 m\(^{-1}\). The wheels of the robot provide sufficient friction with the ground to make the rolling without slipping assumption implicitly made in (1) hold. However, the steering linkage to front wheels permits up to \( \pm 7 \) degrees of error. This error source is not captured by the mathematical model (1) used for control design. The chassis of the robot measures \( 30 \times 22 \times 20 \) cm (l/w/h) and is controlled from an Intel Atom Notebook. Onboard electronics provide low-level commands to the motors while the proposed control algorithm is implemented on the notebook, hereafter called the control computer, running the Robot Operating System (ROS) in Linux.

To implement Algorithm 2, all of the robot’s states are needed. To this end, an Indoor Positioning System (IPS) is employed using the NaturalPoint OptiTrack local positioning system. The IPS uses sixteen near-infra red cameras. Infra red (IR) reflectors are attached to the robot’s chassis to make the position \( (x_1, x_2) \) and orientation \( x_3 \) available for feedback, via the IPS, over WiFi. The control computer uses multithreaded Publish/Subscribe model to read the position and orientation of the robot at 100Hz from the IPS.

In many car-like robot platforms, the steering angle can be directly measured using a potentiometer or an absolute optical encoder; however the Chameleon R100 lacks this feature. Since the steering angle cannot be measured by the IPS, a standard Extended Kalman Filter (EKF) is used to obtain estimate \( \{x_1, x_2, x_3, x_4, x_5, x_6\} \) from the measurements \( \{x_1, x_2, x_3\} \) and the control inputs \( \{u_1, u_2\} \). The control inputs of the Chameleon are its steering angle and translational speed. However, the control inputs of (1) are the rate of change of the steering angle and translational speed. The steering control input can be computed from the rate of change of steering angle by integration.

![The Chameleon R100 robot.](image1.png)

(a) The Chameleon R100 robot.

![Experimental Setup.](image2.png)

(b) Experimental Setup.

**TABLE I: Controller gains used in Section VI**

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbols</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transversal gains (16)</td>
<td>{k_1, k_2, k_3}</td>
<td>{-46.3, -38.7, -10.8}</td>
</tr>
<tr>
<td>Tangential gains (17)</td>
<td>{k_4, k_5, k_6}</td>
<td>{0, -1.3, -2.3}</td>
</tr>
</tbody>
</table>
B. Experimental Results

In the first experiment the Chameleon R100 robot is asked to follow a circular path of radius \( r = 1.3 \) meters while maintaining a constant speed of \( \eta_{2}^{\text{ref}}(t) = 0.3 \) m/sec along the path, i.e., \( \eta^{\text{ref}}(t) = (0, 0.3, 0) \) which clearly satisfies Assumption 4.

In the experiment, pole placement was used to select the gains so that the control signals did not saturate. The closed-loop transversal dynamics were designed to converge to zero faster than the closed-loop tangential error dynamics, to promote convergence to the path over progress along the path. In the following experiments, the desired pole locations for the transversal states \( \xi \) were chosen as \(-3.9, -3.6, -3.3\). The desired pole location for the tangential error states \( e_\eta = (\eta_2-\eta_{2}^{\text{ref}}, \eta_3-\eta_{3}^{\text{ref}}) \) are chosen to be \(-1.2, -1.1\). The controller gains computed for the desired pole locations as shown in the Table 1.

The robot’s initial position is indicated by a solid green dot in Figure 2(a). The desired circle is represented by a dotted line in the figure.

![Fig. 2: Chameleon R100 following the circular path.](image)

![Fig. 2: Chameleon R100 following the circular path.](image)

(a) Chameleon R100 following the circular path.

(b) Convergence of \( \xi_1, \xi_2, \xi_3 \) states.

Fig. 2: Chameleon R100 robot following the circular curve \( \sigma : [0, 2.6\pi) \rightarrow \mathbb{R}^2, \lambda \mapsto \text{col}(1.3\sin(\lambda/1.3), 1.3\cos(\lambda/1.3)) \).

The position of the robot along the path is given by the transformed state \( \eta_1 \in [0, 2.6\pi) \). In this example the path is closed and has arc-length \( 2.6\pi \); therefore \( \mathcal{D} = [0, 2\pi r) = [0, 2.6\pi) \) and \( \eta_1 \) remains bounded between 0 to \( 2\pi r \) as shown in Figure 3(a). The tangential state \( \eta_2 \) is shown in Figure 3(b). A fixed tangential speed of 0.3 m/sec was chosen due to the limited capabilities of the vehicle and the limited test area available inside the indoor positioning system capture region. Simulation examples with variable speed profiles can be found in [12].

![Fig. 3: Velocity and position of the Chameleon R100 robot while following the circular curve \( \sigma : [0, 2.6\pi) \rightarrow \mathbb{R}^2, \lambda \mapsto \text{col}(1.3\sin(\lambda/1.3), 1.3\cos(\lambda/1.3)) \).](image)

In the second experiment the Chameleon R100 robot is made to follow a non-closed sinusoidal path. Figure 4(a) shows that the robot first converges to the desired path and follows it. Due to limited lab space the robot is asked to follow only a small portion of the sinusoidal path. All the transversal states (the \( \xi \) states) converge to zero (Figure 4(b)). As the robot follows the sinusoid path a desired speed of 0.3 m/sec is achieved as shown in Figure 5.

In the third experiment the repeatability of the proposed controller is tested on a circular path of radius 1.3 meters. The experiment is repeated six times and the convergence of the path following error is analyzed. In each test the robot converges to the desired path starting from an initial point away from the path as shown in Figure 6(a). The path following error \( e_{PF} := \sqrt{x_1^2 + x_2^2} - 1.3 \), is shown in Figure 6(b). The initial pose (position and orientation) and steady-state path following error \( |e_{PF}^\text{ss}| := \lim_{t \to \infty} \sup |e_{PF}| \)
of the robot in each run is presented in Table II. Figure 7 gives a zoomed in view of the path following error. We see that the path following error in each run remains within \( \pm 0.015 m \). We conclude that path following controller gives fairly accurate and reliable results as the mean path following error of the six runs is 1.0689cm with a standard deviation of 0.154cm.

We observed that the closed-loop performance is very sensitive to IPS calibration errors. A small misalignment between the center of IR reflectors and center of the rear axle, i.e., \((x_1,x_2)\), is reflected in the path following error. Moreover, we observed that the error is reduced by a few centimeters if an EKF is used, as described above, on all six states of the system. An adaptive path following controller may perform better in the face of calibration errors.
TABLE II: Steady-state path following error. The initial conditions \((x_1(0), x_2(0))\) and \(x_3(0)\) are given in metres and radians, respectively. The path following error is given in centimetres.

| Test | \((x_1(0), x_2(0))\) | \(x_3(0)\) | \(|e_{PF}|\) |
|------|----------------------|------------|-----------|
| 1    | (3.0267, 0.4083)     | 1.8153     | 1.0580    |
| 2    | (−0.1075, −1.7628)  | 0.1440     | 1.3706    |
| 3    | (2.7383, 1.2309)     | 2.3205     | 0.9556    |
| 4    | (1.4719, 1.8907)     | 2.9793     | 1.0089    |
| 5    | (−0.0971, −0.3565)  | −0.6987    | 1.0148    |
| 6    | (−2.2894, −0.4131)  | −1.0454    | 0.9992    |

VII. CONCLUSIONS AND FUTURE RESEARCH

In this paper a path following controller is designed for the kinematic model of a car-like mobile robot using transverse feedback linearization with dynamic extension for a large class of paths. The control method is experimentally demonstrated on a Chameleon R100 Ackermann steering robot. It has been shown that the path following controller forces the robot to converge and then follow the desired path with very small error. Future research includes precise characterizations of the region of attraction of the proposed controllers, and the use of adaptive, nonlinear PI path following controllers based on the notion of immersion and invariance \[34\] to reduce sensitivity to sensor calibration errors.

REFERENCES

Adeel Akhtar received the M.A.Sc. degree from the University of Waterloo, Waterloo, ON, Canada, in 2011. He received his B.E. degree from National University of Science and Technology, Pakistan. In 2011, he joined Waterloo Autonomous Vehicles Lab (WAVELab) and is currently working towards a Ph.D. in the Department of Mechanical and Mechatronics Engineering at the University of Waterloo in Waterloo, ON, Canada. His research interests include nonlinear control and estimation of autonomous vehicles.

Christopher Nielsen received his B.A.Sc. in Electrical Engineering from the University of Waterloo in 2002. He received his M.A.Sc. and Ph.D. degrees from the University of Toronto in 2004 and 2008 respectively. In 2008 he joined the Department of Electrical and Computer Engineering at the University of Waterloo where he is an Associate Professor. Current research interests include geometric control theory, motion control, and output feedback.

Steven L. Waslander (M’05) received his Ph.D. from Stanford University in 2007, his M.S. from Stanford University in 2002, both in Aeronautics and Astronautics, and his B.Sc.E. in 1998 from Queen’s University. He has worked as a Control Systems Analyst for Pratt & Whitney Canada from 1998 to 2001. In 2008, he joined the Department of Mechanical and Mechatronics Engineering at the University of Waterloo in Waterloo, ON, Canada, as an Assistant Professor. He is the Director of the Waterloo Autonomous Vehicles Laboratory (WAVELab). His research interests are in the areas of multi-agent control and coordination, air traffic management and autonomous mobile robotics, specifically quadrotor helicopters.