Hindawi Publishing Corporation
Journal of Robotics
Volume 2015, Article ID 919073, 15 pages
http://dx.doi.org/10.1155/2015/919073

Research Article

Bioinspired Tracking Control of High Speed Nonholonomic Ground Vehicles

Adam Shoemaker and Alexander Leonessa

Center for Dynamic Systems Modeling and Control, Department of Mechanical Engineering, Virginia Tech, Blacksburg, VA 24061, USA

Correspondence should be addressed to Adam Shoemaker; shoe@vt.edu

Received 5 January 2015; Revised 19 May 2015; Accepted 24 June 2015

Academic Editor: Maki K. Habib

The behavior of nature's predators is considered for designing a high speed tracking controller for nonholonomic vehicles, whose dynamics are represented using a unicycle model. To ensure that the vehicle behaves intuitively and mimics the biologically inspired predator-prey interaction, saturation constraints based on Ackermann steering kinematics are added. A new strategy for mapping commands back into a viable envelope is introduced, and the restrictions are accounted for using Lyapunov stability criteria. Following verification of the saturation constraints, the proposed algorithm was implemented on a testing platform. Stable trajectories of up to 9 m/s were achieved. The results presented show that the algorithm demonstrates significant promise in high speed trajectory tracking with obstacle avoidance.

1. Introduction

With the development of driverless technologies, there is an increasing demand for high speed systems capable of dealing with unstructured environments. Many commonly used obstacle avoidance methods focus on generating a path and forcing the vehicle directly to the desired trajectory [1–7]. Unfortunately, in many cases, such as potential field methods, the desired route is rarely smooth and does not sufficiently account for the desired velocity or vehicle capability [3, 8–13].

To find a more intuitive behavior, we look to biology for inspiration, specifically a generic predator-prey interaction. A cheetah chasing its prey, for example, does not directly mimic the path of its target. Instead, it creates a relatively smooth path, although its target may be moving somewhat chaotically [14]. The smooth path, in turn, allows the cheetah to maintain much of its high speed and still meet the desired goal [14]. As will be shown, the proposed algorithm mimics this behavior by introducing a reference system for the vehicle to follow, analogous to how a predator chases its prey.

Due to the high speed nature of our goal, this study focuses on Ackermann steering based platforms. Because of its simplicity and ability to capture nonholonomic constraints, we begin by considering a unicycle model, which will be shown to fit within the confines of an Ackermann platform by adding saturation constraints. While there exists extensive research in the control of unicycle type robotic systems [15–21], the study begins with the framework laid out in [21, 22] as it introduces a relationship similar to the predator-prey interaction. In particular, when considering the path of the cheetah, its trajectory can be thought of as a filter to the prey's path; this is possible due to the cheetah maintaining a certain following distance from its prey. The larger the distance is, the more filtered the path becomes. In [22], the authors provide a control algorithm that mimics this behavior. By controlling velocity and angular rate, an algorithm is designed in [22] which theoretically tracks a virtual target at a time-varying distance. However, while capturing certain constraints, the unicycle model, and by consequence the work of [22], are not comprehensive. Much as the cheetah is unable to immediately turn independently of speed, neither are many ground vehicles. This paper focuses on extending preceding work to better satisfy the desired bioinspired model.

This work is organized as follows. In Sections 2 and 3 we introduce the algorithm laid out in [22] along with additional restrictions placed on the system structure and following distance parameters, which are aimed at bolstering controller

reliability. Moreover, we evaluate several limitations of the base algorithm in the context of the biologically inspired predator-prey relationship and high speed ground vehicles. In Section 4, we introduce constraints on angular rate and velocity commands, which are based on Ackermann steering platform kinematics. In this section, we develop a new strategy for mapping commands into a viable control space that reflects these kinematic restraints. We additionally provide constraints on the following distance parameters to ensure stability. The algorithm is shown to satisfy the Lyapunov stability criteria, while still meeting all of these requirements.

The proposed algorithm was ultimately implemented on a testing platform, the results of which are presented in Section 5. To satisfy high-level commands, low-level PI controllers were used to control commanded velocity and angular rate. The reference system was manipulated using a basic potential field methodology of strictly attractors and repulsors similar to that presented in [23–25]. Using an Extended Kalman Filter to provide reliable state information, the full algorithm was used to achieve stable trajectories up to 9 m/s, the top speed of the platform. Further work will focus on accounting for actuator dynamics and extending the method to multiple platforms.

2. System Definition

We begin with a unicycle model as it adequately captures the nonholonomic constraints of our Ackermann steering platform. Moreover, the simplicity of the model directly lends itself to controller design. While typically used for differential drive vehicles, we note that this relationship is not exclusive, and by developing saturation constraints, we can force the unicycle model to operate in the confines of an Ackermann steering platform. The unicycle model used is as follows,

$$\dot{p}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \end{bmatrix}, \tag{1}$$

$$\dot{\theta}(t) = \omega(t), \tag{2}$$

where $p(t) \triangleq [x(t) \;\; y(t)]^{\mathrm{T}} \in \mathbb{R}^2$ is the vehicle's position, $v(t) \in \mathbb{R}$ is the longitudinal velocity, $\theta(t) \in \mathbb{R}$ is the heading, and $\omega(t) \in \mathbb{R}$ is the vehicle's angular velocity. A reference system is then introduced to represent a virtual target,

$$p_{\mathrm{r}}(t) \triangleq \begin{bmatrix} x_{\mathrm{r}}(t) \\ y_{\mathrm{r}}(t) \end{bmatrix}, \tag{3}$$

where $p_{\mathrm{r}}(t) \in \mathbb{R}^2$ is the reference system's position. In the interest of continuous controller commands, we must guarantee that the reference system is composed of class C^1 functions in time. For convenience, the linear velocity of the reference system is defined as

$$v_{\mathrm{r}}(t) \triangleq \sqrt{\dot{x}_{\mathrm{r}}(t)^2 + \dot{y}_{\mathrm{r}}(t)^2}, \tag{4}$$

where $v_{\mathrm{r}}(t) \in \mathbb{R}$.

In the following section, we will see that the controller design will guarantee the system described by (1) and (2) will converge to a sufficiently small neighborhood around the reference system, described by (3), such that $\|p(t) - p_{\mathrm{r}}(t)\| \to d^*(t)$, as $t \to \infty$, where $d^*(t) > 0$ is a user defined nominal following distance. By guaranteeing that the vehicle follows the reference system in this manner, we enforce our bioinspired predator-prey interaction mentioned previously.

3. Control Design

The overall controller is presented in two distinct steps. The first step involves the basic design, while the second step includes saturation algorithms based on Ackermann steering kinematics. In this first step, we assume that the reference system can be controlled to follow a desired trajectory. By ensuring that the actual system converges to a small neighborhood of the reference system, we ensure that the system follows the desired trajectory as well. At the same time, a sufficiently large separation between reference system and vehicle allows a buffer for the vehicle to track the reference, without violating its nonholonomic constraints.

In order to properly design the following distance, it is necessary to define the error between the vehicle and reference system. We start by assuming that the origins of the body fixed coordinate frame, $\mathscr{B}$, and of the global inertial coordinate frame, $\mathscr{U}$, coincide with the center of mass of the vehicle in the horizontal plane. An orthonormal transformation from $\mathscr{B}$ to $\mathscr{U}$ is then defined,

$$R(\theta(t)) \triangleq \begin{bmatrix} \cos(\theta(t)) & -\sin(\theta(t)) \\ \sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix}. \tag{5}$$

Using this transformation along with (1) and (3), the position error can be expressed in the body coordinate frame as

$$e(t) \triangleq R^{\mathrm{T}}(\theta(t))\left(p_{\mathrm{r}}(t) - p(t)\right), \tag{6}$$

where $e(t) \in \mathbb{R}^2$. This position error can be thought of as a vector expression of longitudinal and lateral error in the body frame. Next we introduce the commanded distance vector,

$$\delta(t) \triangleq \begin{bmatrix} d(t) \\ 0 \end{bmatrix}, \tag{7}$$

where $d(t) > 0$ is the commanded following distance. By guaranteeing that $e(t) \to \delta$ as $t \to \infty$, the longitudinal error will converge to $d(t)$ while the lateral error goes to zero.

To provide this behavior, we introduce the following control law ([22]),

$$\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = \Delta^{-1}(t) \cdot \left(K\tanh(e(t) - \delta(t)) + R^{\mathrm{T}}(\theta(t))\,\dot{p}_{\mathrm{r}}(t) - \dot{\delta}(t)\right), \tag{8}$$

where

$$\Delta(t) \triangleq \begin{bmatrix} 1 & 0 \\ 0 & d(t) \end{bmatrix}, \qquad K \triangleq \begin{bmatrix} k_v & 0 \\ 0 & k_\omega \end{bmatrix}, \tag{9}$$

and $k_v > 0$ and $k_\omega > 0$ are scalar tuning constants used to provide bounded signals to the commanded longitudinal and angular velocities, respectively. This control law, which is more thoroughly derived in [22], is based on Lyapunov stability theory. In [22], the authors show it to be the natural result that guarantees a negative definite Lyapunov time derivative, given the Lyapunov function (12) introduced in the following theorem.

In order to ensure that $\Delta^{-1}(t)$ is defined for all $t \geq 0$, we must guarantee that $d(t) \neq 0$ for all $t \geq 0$. In practice, if $d(t) < 0$, the vehicle will follow the reference system but in reverse, such that $v(t) < 0$. As such, we restrict the commanded following distance to $d(t) > 0$. For design purposes, a more rigorous constraint of $d(t) > (\beta - \varepsilon) > 0$, for all $t \geq 0$, is enforced, where $\beta > \varepsilon > 0$ are tuning parameters used to place a minimum bound on $d(t)$. The commanded following distance, $d(t)$, is designed such that $d(t) \to d^*(t)$ as $t \to \infty$, provided that $d^*(t) \geq \beta$ as well.

At this point, we note the intuitive form of (8). For instance, from the $R^{\mathrm{T}}(\theta(t))\dot{p}_{\mathrm{r}}(t)$ term, we see that the reference velocity, projected along the vehicle's longitudinal axis, positively contributes to the commanded velocity, $v(t)$. Likewise the reference velocity, projected along the lateral axis, contributes to the commanded angular rate, $\omega(t)$. In examining the $\dot{\delta}(t)$ term, we see that an increasing distance negatively impacts the commanded velocity, as would be expected. Lastly, with regard to the $K\tanh(e(t) - \delta(t))$ term, we see that the difference between longitudinal error and the commanded distance, $d(t)$, contributes positively to the commanded velocity. Along the same lines, lateral error contributes positively to the angular rate command.

With regard to the $\tanh(\cdot)$ term of (8), which is a componentwise operation, we note that while it is unnecessary for theoretical stability, it adds a layer of tuning to the controller. This layer proves particularly useful in experimentation. The function is chosen for its natural saturation capability. Based on the constants k_v and k_ω, the function directly contributes bounded signals to commanded longitudinal and angular velocities, which are based on the difference between the tracking error $e(t)$ and distance vector $\delta(t)$.

Theorem 1. *Consider the system described by (1) and (2), the reference system described by (3), and the feedback controller described by (8). If the nominal distance is restricted such that $d^*(t) \geq \beta$, for all $t \geq 0$, and the distance $d(t)$ is updated according to*

$$\dot{d}(t) = \begin{cases} \Gamma(t), & d(t) \geq \beta \\ \Gamma(t) + \dfrac{\beta - d(t)}{d(t) - (\beta - \varepsilon)}, & d(t) < \beta, \end{cases} \qquad d(0) \geq \beta, \tag{10}$$

where

$$\Gamma(t) \triangleq \dot{d}^*(t) - \lambda\left(d(t) - d^*(t)\right), \tag{11}$$

with tuning constant $\lambda > 0$, then the distance $d(t)$, between the vehicle and reference system, converges to the desired distance, $d^*(t)$, while guaranteeing that $d(t) > (\beta - \varepsilon) > 0$. Meanwhile, the tracking error $e(t)$ converges to the distance vector $\delta(t)$.

Proof. Consider the Lyapunov function candidate

$$V(t) = \frac{1}{2}e_1^{\mathrm{T}}(t)\,e_1(t) + \frac{1}{2}\left(d(t) - d^*(t)\right)^2, \tag{12}$$

where

$$e_1(t) \triangleq e(t) - \delta(t). \tag{13}$$

In order to examine the derivative of the Lyapunov function along the system trajectories, the time derivative of the tracking error, $e(t)$, given by (6), is computed as follows,

$$\begin{aligned} \dot{e}(t) = {} & \dot{R}^{\mathrm{T}}(\theta(t), \omega(t))\left(p_{\mathrm{r}}(t) - p(t)\right) \\ & + R^{\mathrm{T}}(\theta(t))\left(\dot{p}_{\mathrm{r}}(t) - \dot{p}(t)\right). \end{aligned} \tag{14}$$

The time derivative of the orthonormal transformation, $R(\theta(t))$, is given by

$$\begin{aligned} & \dot{R}(\theta(t), \omega(t)) \\ & = \begin{bmatrix} -\omega(t)\sin(\theta(t)) & -\omega(t)\cos(\theta(t)) \\ \omega(t)\cos(\theta(t)) & -\omega(t)\sin(\theta(t)) \end{bmatrix} \\ & = R(\theta(t))\,S(\omega(t)), \end{aligned} \tag{15}$$

where

$$S(\omega(t)) \triangleq \begin{bmatrix} 0 & -\omega(t) \\ \omega(t) & 0 \end{bmatrix}. \tag{16}$$

By substituting (1) and (15) into (14), the error dynamics can be simplified to

$$\begin{aligned} \dot{e}(t) = {} & S^{\mathrm{T}}(\omega(t))\,R^{\mathrm{T}}(\theta(t))\left(p_{\mathrm{r}}(t) - p(t)\right) \\ & + R^{\mathrm{T}}(\theta(t))\,\dot{p}_{\mathrm{r}}(t) \\ & - R^{\mathrm{T}}(\theta(t))\begin{bmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \end{bmatrix} \\ = {} & -S(\omega(t))\,e(t) + R^{\mathrm{T}}(\theta(t))\,\dot{p}_{\mathrm{r}}(t) - \begin{bmatrix} v(t) \\ 0 \end{bmatrix} \\ = {} & -S(\omega(t))\,e(t) + R^{\mathrm{T}}(\theta(t))\,\dot{p}_{\mathrm{r}}(t) \\ & - \begin{bmatrix} v(t) \\ d(t)\,\omega(t) \end{bmatrix} + \begin{bmatrix} 0 \\ d(t)\,\omega(t) \end{bmatrix} \\ = {} & -S(\omega(t))\,e_1(t) - \Delta(t)\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \\ & + R^{\mathrm{T}}(\theta(t))\,\dot{p}_{\mathrm{r}}(t). \end{aligned} \tag{17}$$

By substituting the control law given in (8), the error dynamics are then further simplified to

$$\dot{e}(t) = -S(\omega(t))\,e_1(t) - K\tanh\left(e_1(t)\right) + \dot{\delta}(t). \tag{18}$$

The error dynamics given by (18) are then substituted into the time derivative of the Lyapunov function,

$$\dot{V}(t) = e_1^{\mathrm{T}}(t)\,\dot{e}_1(t) + \left(d(t) - d^*(t)\right)\left(\dot{d}(t) - \dot{d}^*(t)\right). \quad (19)$$

If we consider the case in which $d(t) \geq \beta$, we then substitute (10), (11), and (18) into (19) to obtain

$$\begin{aligned}\dot{V}(t) &= e_1^{\mathrm{T}}(t)\left(-S(\omega(t))\,e_1(t) - K\tanh\left(e_1(t)\right)\right) \\ &\quad - \lambda\left(d(t) - d^*(t)\right)^2 \\ &= -e_1^{\mathrm{T}}(t)\,K\tanh\left(e_1(t)\right) - \lambda\left(d(t) - d^*(t)\right)^2.\end{aligned} \quad (20)$$

From (20), it is trivial to show that again $\dot{V}(t) \leq 0$ for all $t \geq 0$, $d(t) \geq \beta$.

The Lyapunov time derivative is again examined when $d(t) < \beta$. Again, we substitute (10), (11), and (18) into (19) to obtain

$$\begin{aligned}\dot{V}(t) &= -e_1^{\mathrm{T}}(t)\,K\tanh\left(e_1(t)\right) + \left(d(t) - d^*(t)\right) \\ &\quad \cdot\left(-\lambda\left(d(t) - d^*(t)\right) + \frac{\beta - d(t)}{d(t) - (\beta - \varepsilon)}\right) \\ &= -e_1^{\mathrm{T}}(t)\,K\tanh\left(e_1(t)\right) - \lambda\left(d(t) - d^*(t)\right)^2 \\ &\quad + \left(d(t) - d^*(t)\right)\left(\frac{\beta - d(t)}{d(t) - (\beta - \varepsilon)}\right).\end{aligned} \quad (21)$$

The commanded following distance, $d(t)$, is restricted such that $d(t) > (\beta - \varepsilon)$ by design. This behavior is the result of $(\beta - d(t))/(d(t) - (\beta - \varepsilon)) \to \infty$ as $d(t) \to (\beta - \varepsilon)$. As $d(t)$ decreases below β but still remains larger than $(\beta - \varepsilon)$, the additional corrective term pushes $d(t)$ back toward β, eventually guaranteeing $\dot{d}(t) > 0$ as $d(t) \to (\beta - \varepsilon)$. As such, the corrective term $(\beta - d(t))/(d(t) - (\beta - \varepsilon)) > 0$ for $d(t) < \beta$. Additionally, since $d^*(t) \geq \beta > d(t)$, we find that $(d(t) - d^*(t))((\beta - d(t))/(d(t) - (\beta - \varepsilon))) < 0$. Since the remainder of the Lyapunov time derivative terms are the same as given in (20), we can conclude that $\dot{V} \leq 0$ for all $t \geq 0$. □

3.1. Simulation Results. The behavior of the control algorithm is examined prior to the introduction of saturation constraints on commanded longitudinal and angular velocities. Before simulation, it is necessary to choose a nominal following distance. By choosing

$$d^*(t) = \alpha v_{\mathrm{r}}(t) + \beta, \quad (22)$$

for $\alpha > 0$ and $\beta > 0$, we enforce a behavior that tends toward the bioinspired predator-prey model discussed earlier. For example, with higher reference velocities, we intuitively allow a greater distance to respond to instantaneous changes in reference direction. Likewise, (22) draws the vehicle closer to tightly follow slow trajectories. Analogously, large separation allows for a cheetah to easily maneuver while maintaining high speed, whereas a small separation lends itself to tighter tracking.

Table 1: Parameters for simulation without saturation.

Desired Reference Trajectory	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$	$k_v = 1$	$x_{\mathrm{r}}(0) = 0$
$r_y(t) = 10\sin(0.5t)$	$k_\omega = 1$	$y_{\mathrm{r}}(0) = 0$
	$\lambda = 1$	$x(0) = -0.1$
	$\alpha = 0.5$	$y(0) = 0$
	$\beta = 0.1$	$\theta(0) = 0$
		$d(0) = 0.1$

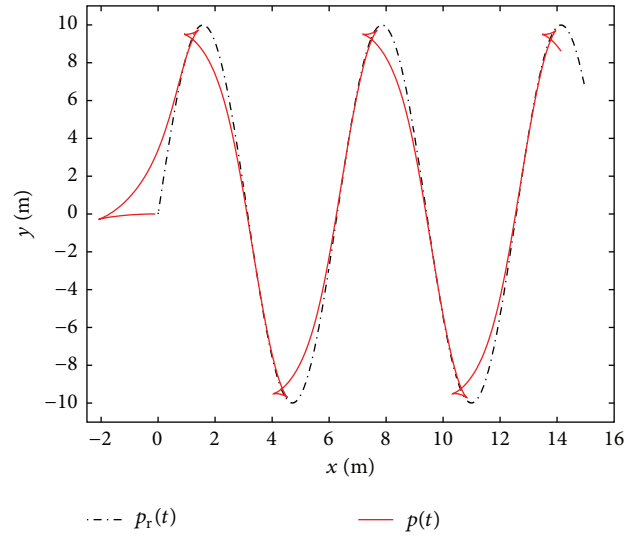

Figure 1: Sinusoidal trajectory without saturation.

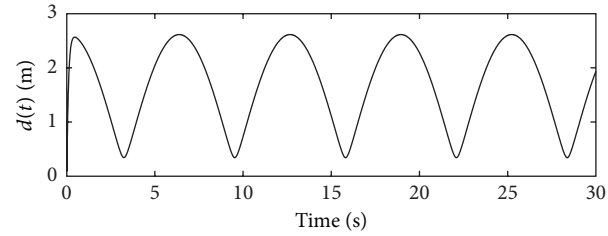

Figure 2: Following distance without saturation.

To evaluate the performance of the controller, we consider the situation in which the reference system is defined as two decoupled first-order systems,

$$\dot{p}_{\mathrm{r}}(t) = \begin{bmatrix} -10x_{\mathrm{r}}(t) + 10r_x(t) \\ -10y_{\mathrm{r}}(t) + 10r_y(t) \end{bmatrix}, \quad (23)$$

where $r_x(t)$ and $r_y(t)$ are the desired reference trajectories in x and y, respectively. Figures 1–3 show the results of simulation using the parameters given in Table 1.

Figure 1 shows the theoretical performance of the system. As intended, the vehicle converges tightly to the reference system's path, while maintaining a varying following distance, shown in Figure 2. Further inspection shows that the minimal distance coincides with the sinusoidal peaks in the trajectory.

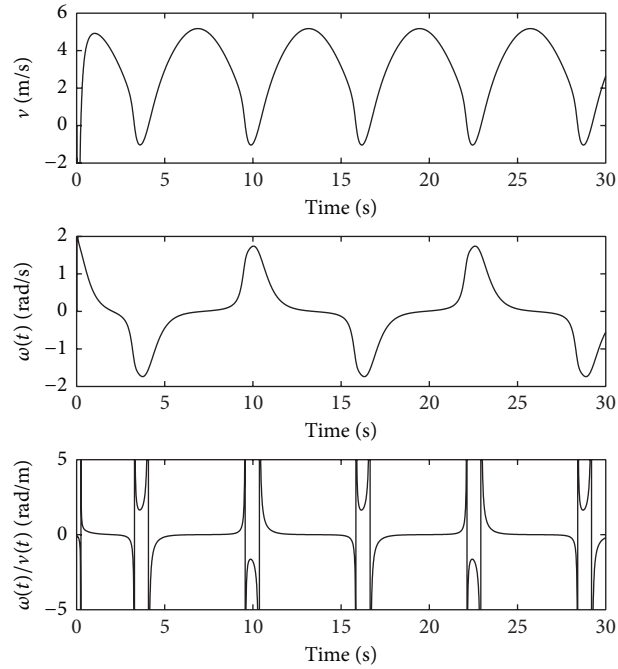

Figure 3: Controller commands for sinusoidal trajectory without saturation.

This relationship is a result of lower reference velocity in the y direction, which leads to close tracking at low speeds around the peaks of the sine wave.

With regard to the controller commands, seen in Figure 3, the trends are as expected. Since the controller is based on a unicycle model, there is no obvious correlation between velocity and angular rate commands. In fact, many of the largest angular rate commands, with regard to magnitude, correspond to the lowest velocity commands. This relationship is exemplified by the third graph which examines the ratio of angular rate to longitudinal velocity. At multiple instances throughout the simulation, this ratio experiences singularities.

4. Saturation Constraints

Since the control algorithm is geared toward high speed control and obstacle avoidance, we consider an Ackermann steering platform for testing and implementation. However, due to the behavior discussed in the previous section, where the ratio of angular to longitudinal velocity becomes singular, the control algorithm is not suitable for an Ackermann steering platform without modification.

Additionally, we observe in Figure 3 that the controller naturally commands negative velocities. Specifically, at the start of simulation, the vehicle immediately backs up to satisfy the error and distance requirements, which for this case are largely dictated by $\dot{\delta}(t)$. Further instances of negative velocity occur throughout the trajectory while undergoing tight cornering. In practice, however, we would like the behavior to better follow that of a predator chasing its prey. In the case of the cheetah, it is unlikely that the creature will stop, back up while turning, and then continue the chase. Instead, it is more intuitive to continue along its current path, preserving speed while correcting heading to the best of its ability.

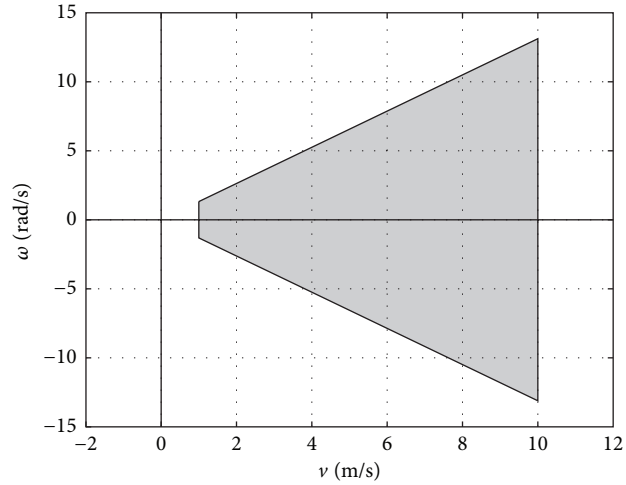

Figure 4: Graphical representation of viable control envelope (shaded).

In order to elicit the same behavior, the following saturation constraints are introduced based on an Ackermann steering platform,

$$\left|\frac{\omega(t)}{v(t)}\right| \leq \frac{\tan(\phi_{\max})}{L}, \tag{24}$$

$$v_{\min} \leq v(t) \leq v_{\max}, \tag{25}$$

where $L > 0$ is the wheelbase length from rear to front axle, $\phi_{\max} > 0$ is the maximum steering angle deflection referenced from the longitudinal body axis, $v_{\min} > 0$ is a lower bound on the longitudinal velocity, and $v_{\max} > 0$ is an upper bound on longitudinal velocity. In this case it is assumed that the maximum steering angle is the same in both directions, allowing $\phi_{\max}$ to be used for both minimum and maximum saturation of the angular rate.

The envelope is graphically demonstrated via the shaded region seen in Figure 4. In Figure 4, we note the minimum bound on velocity, which excludes the origin. As a result, the algorithm is unable to achieve a stationary command. This choice is a product of our bioinspired cheetah model, which we want to continue on a forward trajectory at a minimum velocity bound, rather than stopping or reversing direction. Additionally, if commands were mapped to the origin during saturation, the vehicles position would remain unchanged, potentially giving it no way to exit the saturation condition. To prevent this stagnation, we again enforce a minimum velocity.

4.1. First Stage Saturation. During operation, the control law given in (8) can viably produce a command anywhere in the v-ω plane; however, it is desired to map the commands to somewhere within the given envelope while still preserving directional intent. Moreover, any mapping of commands

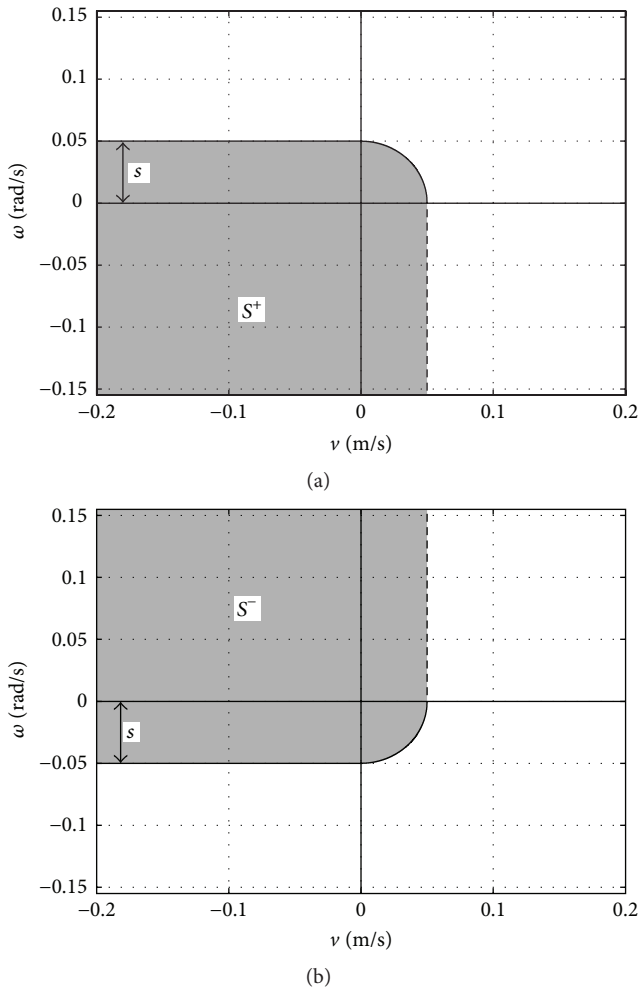

FIGURE 5: Graphical representation of S^+ (a) and S^- (b).

should be continuous, not allowing for instantaneous transitions from one region of the envelope to another. In order to address these constraints, we present a dual stage saturation algorithm.

Based on our choice of regions in the second stage of saturation, which will be discussed in the next section, it is possible for commands to instantaneously change locations on the envelope. Specifically, if the controller commands cross the negative v axis or pass through the origin, an instantaneous change will occur in the second stage mapping, which is undesirable. In order to prevent these potential problems and preserve directional intent, the first stage is used to prevent commands from entering into a small boundary around the negative v axis and the origin. It should be noted that this saturation stage does not result in commands mapped back to the envelope, which is instead achieved by the second stage.

The first stage ensures that controller commands cannot traverse the region shown in Figures 5 and 6, which is defined as

$$S \triangleq S^+ \cap S^-, \tag{26}$$

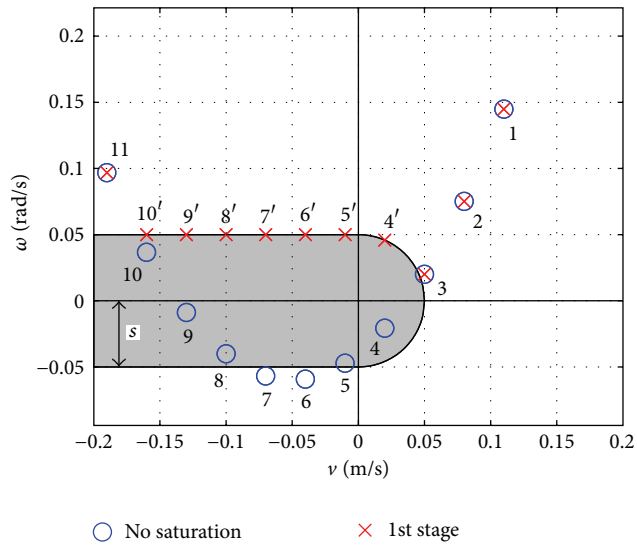

FIGURE 6: First stage representation of S with saturation example.

where

$$S^+ \triangleq \left\{(v,\omega) \in \mathbb{R}^2 : 0 \le \omega \le s, v \le \sqrt{s^2 - \omega^2}\right\} \cup \left\{(v,\omega) \in \mathbb{R}^2 : v < s, \omega < 0\right\}, \tag{27}$$

$$S^- \triangleq \left\{(v,\omega) \in \mathbb{R}^2 : -s \le \omega \le 0, v \le \sqrt{s^2 - \omega^2}\right\} \cup \left\{(v,\omega) \in \mathbb{R}^2 : v < s, \omega > 0\right\}, \tag{28}$$

where $s \in (0, v_{\min})$ defines the size of this small region around the negative v axis. In practice, s can be arbitrarily small as its only purpose is to provide a boundary around the negative v axis. In general, small values, such that $s \ll v_{\min}$, are preferred to provide the least interference with controller intent. Figure 6 gives an example of the saturation implementation, which is detailed in the following discussion.

During implementation, if at time $t_e > 0$ the controller issues a command such that $(v(t_e), \omega(t_e)) \in \partial S$, then $\omega_e \triangleq \omega(t_e)$ is recorded. If $\omega_e \ge 0$, then for all subsequent controller commands such that $(v(t), \omega(t)) \in S^+$ the algorithm maps the commands to $(v(t), \omega_s(t)) \in \partial S^+ \cap S^+$. Likewise in the case of $\omega_e < 0$, for all subsequent commands such that $(v(t), \omega(t)) \in S^-$ the algorithm maps the command to $(v(t), \omega_s(t)) \in \partial S^- \cap S^-$. We note that $\omega_s(t)$, the resultant angular rate command, is defined by the point on the boundary that corresponds to the original velocity command.

Figure 6 demonstrates a parabolic command evolution before and after the first saturation stage. In this case, we consider the commands evolving from right to left in the order they are numbered. The circles represent the original commands from the controller, and the crosses demonstrate the resulting command after the first saturation stage. Examining the figure, we see that commands 1–3 are left unaltered, as they are outside of the region S. At some point between 3 and 4, the command crosses the boundary of S, at which point ω_e is recorded to be of some positive value. Since $\omega_e \ge 0$, all subsequent commands such that $(v(t), \omega(t)) \in S^+$ are mapped

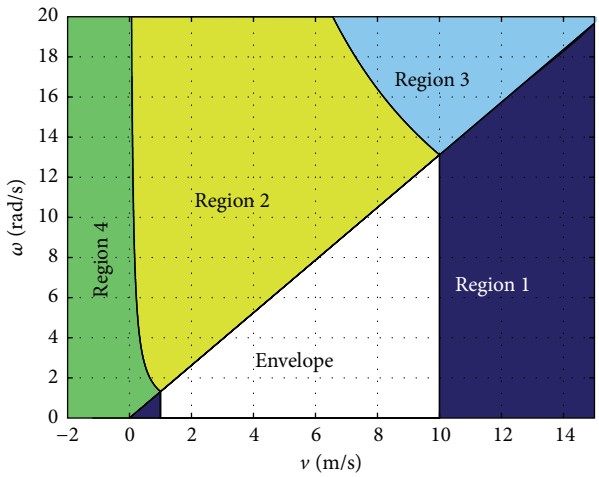

FIGURE 7: Region classification for second stage saturation.

to $\partial S^+ \cap S^+$, while preserving the velocity command, $v(t)$. This trend is the case for commands 4–10 which are mapped to $4'$–$10'$, respectively. Of particular interest are commands 6 and 7. Although these commands exit the region S, they are still contained in S^+. As such, the saturation algorithm continues to map these commands to ∂S^+ using the same methodology. Lastly, we look at command 11, which occurs at some point after the commands have exited S^+. In this case, since $(v(t), \omega(t)) \notin S^+$ the saturation algorithm is no longer active.

4.2. Second Stage Saturation. Once the commands are passed through the first stage of saturation, they are categorized to a specific region based on their location in the v-ω plane. Figure 7 shows four separate regions which use various methodologies to map the commands into the viable envelope. In Region 1, we focus on preserving the radius of curvature of the original commands as this directly relates to the intended path of the controller. Due to the limitations of the Ackermann platform, however, not all curvatures are achievable. As such, in Region 2, the second stage saturation algorithm maps commands to the nearest achievable radius of curvature while preserving lateral acceleration. We choose to preserve lateral acceleration so that the acceleration perceived by the vehicle and its sensors will remain unchanged. In the case of Regions 3 and 4, commands originate outside of any achievable radius of curvature or lateral acceleration. As a result, these commands are mapped to the closest achievable points that preserve intent. As expected, all commands originating within the envelope are left unaltered. For the remainder of this discussion, we only consider cases of $\omega(t) \geq 0$, as the other case is simply a reflection about the v axis.

4.2.1. Region 1 Methodology. Ideally, during saturation it is desirable to reproduce the same trajectories that are commanded by the controller. In order to elicit this behavior, we examine the radius of curvature given by

$$r(t) = \frac{v(t)}{\omega(t)}. \tag{29}$$

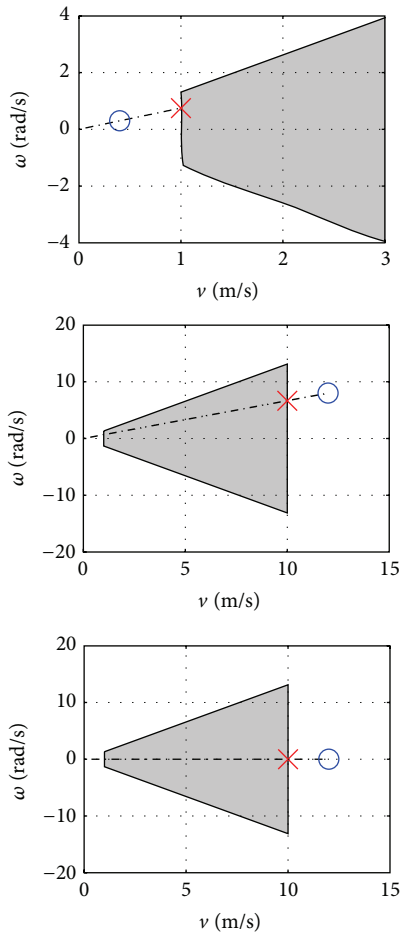

FIGURE 8: Region 1 examples for second stage saturation.

When the requested command lies within Region 1, which is given by the dark blue regions on the left and right sides of the envelope in Figure 7, the second stage saturation algorithm maps the new command to the nearest point on the envelope that provides the same radius of curvature. By maintaining the same radius of curvature, we ensure that the intended path is followed to the best of the vehicles capability. We note that the origin is included as part of Region 1, however, this presents a potential for discontinuous mapping depending on command evolution. For this reason, the first stage saturation method is used to prevent transitions through the origin.

Figure 8 provides several examples of Region 1 saturation. In Figure 8, the circles represent the command received from the first stage of saturation, and the crosses represent the command following the second saturation stage. In the first example, the command is mapped to the minimum velocity while still preserving the $\omega(t)/v(t)$ ratio. The second and third examples exhibit the same behavior, mapping to the maximum velocity instead. Of particular interest is the third example, which maps along the v axis. In this case, the radius of curvature is infinite, but by using the inverse of curvature, the methodology is preserved. For this reason, the reciprocal of the radius is typically used to prevent singularities in computation.

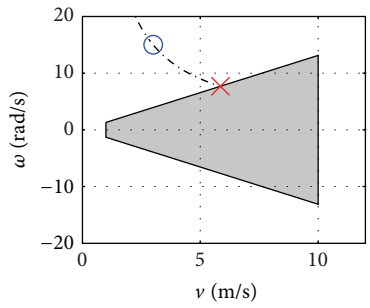

Figure 9: Region 2 example for second stage saturation.

4.2.2. Region 2 Methodology. Due to the constraints of an Ackermann platform, there are a variety of conditions in which the desired radius of curvature is not achievable, as is the case of Region 2. To find a reasonable alternative, we start by considering the lateral acceleration of the vehicle given by

$$a_{\mathrm{L}}(t) = v(t)\,\omega(t). \tag{30}$$

Commands originating in Region 2 are mapped to the point on the envelope that is characterized by the same lateral acceleration. This point also corresponds to the nearest achievable radius of curvature. Figure 9 demonstrates saturation behavior for Region 2. The circle represents the command received from the first stage of saturation, and the cross represents the command following the second saturation stage. In this figure, the black dashed line represents the laws of constant lateral acceleration. As explained, both commands have the same lateral acceleration, with one being in the achievable command space.

4.2.3. Region 3 Methodology. While lateral acceleration offers an intuitive choice for mapping commands to an obtainable radius of curvature, Ackermann platforms inherently have a maximum lateral acceleration, given by

$$a_{\mathrm{L}_{\max}} \triangleq v_{\max}^2 \frac{\tan(\phi_{\max})}{L}. \tag{31}$$

When the requested command exceeds the minimum radius of curvature and the maximum lateral acceleration, the command is mapped to the maximum velocity and maximum turning angle, represented by the corner of the envelope. Region 3 of Figure 7 depicts the area in question. By holding the command at the corner of the envelope in Region 3, we ensure a continuous mapping from Region 2 to Region 3 and from Region 3 to Region 1.

4.2.4. Region 4 Methodology. Although lateral acceleration offers a good methodology for mapping commands, it does not always elicit optimal behavior. Recalling from Section 3.1, the controller can command high angular rates while also commanding little to no longitudinal velocity. In these cases, the lateral acceleration is small. If we were to map to the envelope maintaining this lateral acceleration, the result would yield a forward velocity, with a nearly zero angular rate, which would be opposite to desired. Instead, the vehicle should ideally turn as sharp as possible.

In Region 4, we examine the situation in which the radius of curvature is not obtainable and the command lies below the minimum lateral acceleration threshold given by

$$a_{\mathrm{L}_{\min}} \triangleq v_{\min}^2 \frac{\tan(\phi_{\max})}{L}. \tag{32}$$

In this case, the command is mapped to the minimal velocity and maximum angular rate, characterized by the left corner of the envelope given in Figure 7. For the situations in which the vehicle has a large requested angular rate with small velocity, the algorithm ensures that the vehicle turns as sharp as possible at a minimal velocity until the controller requests more achievable behavior.

We extend Region 4 to encompass all negative velocity commands as well since the goal is to turn as much as possible until the controller requests positive velocities. We note that while Region 4 includes portions of the first stage of saturation, these portions will not be reached due to the effects of the first stage of saturation. If the controller commands were to cross the v axis while the requested velocity is negative, indicated by Figures 5 and 6, the second stage saturation algorithm would fully reverse the steering command. However, the first level of saturation restricts this potential chattering behavior by keeping the command in the same quadrant until the controller requests positive velocities.

4.3. Maintaining Lyapunov Criteria. For the above cases, the controller commands are altered without regard to how it affects the tracking performance of the overall system. In order to ensure that the asymptotic stability of the error dynamics is still achieved, the control law is backsolved from (8) as ([22])

$$\begin{aligned}\dot{p}_{\mathrm{r}}^{\mathrm{sat}}(t) = R(\theta(t))\Bigg(&\Delta(t)\begin{bmatrix} v^{\mathrm{sat}}(t) \\ \omega^{\mathrm{sat}}(t)\end{bmatrix} \\ &- K\tanh(e(t)-\delta(t)) + \dot{\delta}(t)\Bigg),\end{aligned} \tag{33}$$

where $\dot{p}_{\mathrm{r}}^{\mathrm{sat}}(t) \in \mathbb{R}^2$ is the resulting reference time derivative, and, in this case, $v^{\mathrm{sat}}(t)$ and $\omega^{\mathrm{sat}}(t)$ are determined from the saturation algorithm. It is important to realize that $d^*(t)$ must be at least a class C^1 function in time to guarantee both that $\dot{d}(t)$ exists and also that the control law is continuous. To elicit this behavior, we modify the nominal following distance to be

$$\ddot{d}^*(t) + 2\zeta_d\omega_d\dot{d}^*(t) + \omega_d^2 d^*(t) = \omega_d^2 d_{\mathrm{ref}}(t), \tag{34}$$

where $\zeta_d > 0$, $\omega_d > 0$ are tuning constants and $d_{\mathrm{ref}}(t)$ is the user desired reference distance.

4.4. Simulation Results with Saturation. Prior to saturation, we chose $d^*(t)$ to follow the relationship given by (22). In order to obtain similar behavior during saturation and adhere

Table 2: Parameters for simulation with saturation.

Desired Reference Trajectory	Tuning Parameters	Initial Conditions
$r_x(t) = 0.5t$	$k_v = 1$	$x_r(0) = 0$
$r_y(t) = 10\sin(0.5t)$	$k_\omega = 1$	$y_r(0) = 0$
	$\lambda = 1$	$x(0) = -0.1$
	$\alpha = 0.5$	$y(0) = 0$
	$\beta = 0.1$	$\theta(0) = 0$
	$L = 0.3556$	$d(0) = 0.1$
	$\gamma_{\max} = 25^\circ$	$d^*(0) = 0.1$
	$v_{\min} = 1$	$\dot{d}^*(0) = 0$
	$v_{\max} = 10$	
	$s = 0.01$	
	$\omega_d = 2.5$	
	$\zeta_d = 0.85$	

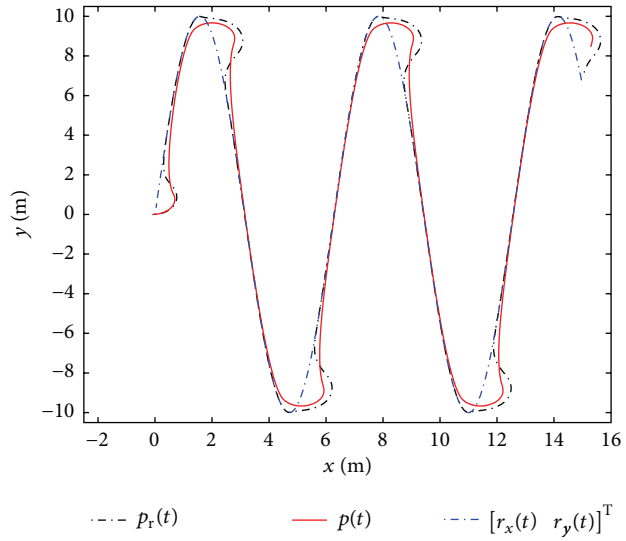

Figure 10: Sinusoidal trajectory with saturation algorithm.

to the criteria of the previous section, we choose a desired reference distance of

$$d_{\text{ref}}(t) = \alpha v_r(t) + \beta. \tag{35}$$

For comparison purposes, we introduce the same reference trajectory of (23). All simulation parameters are listed in Table 2.

Figure 10 demonstrates the resulting trajectory information with the addition of saturation constraints. Overall, the vehicle still follows the trajectory given by $[r_x(t)\ \ r_y(t)]^T$ of Table 2, which is desired. However, in the case of sharp turns, the vehicle continues forward, locking itself into a circular trajectory until it is able to reobtain the desired path. To facilitate this behavior, the reference system is pushed away from the trajectory detailed by $[r_x(t)\ \ r_y(t)]^T$, overriding the dynamics described by (23). In addition to following the path, the saturation algorithm also eliminated the prior initial behavior, characterized by an immediate reversal of trajectory

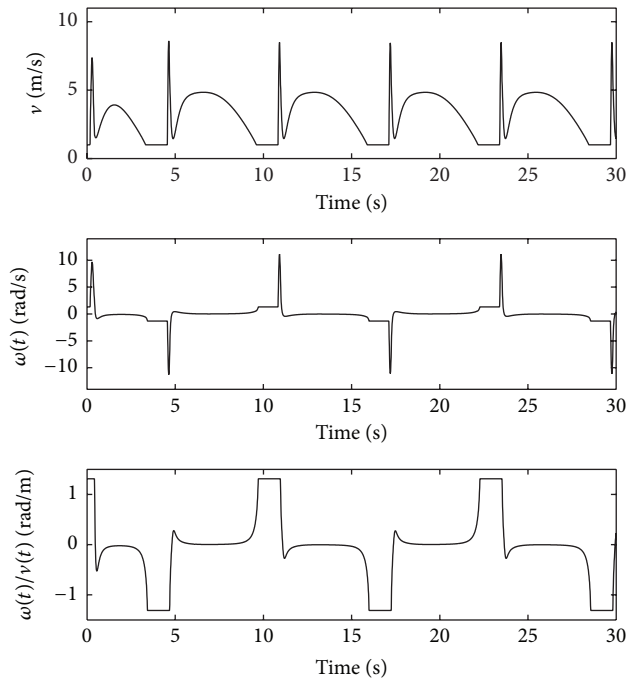

Figure 11: Controller commands for sinusoidal trajectory with saturation.

to satisfy distance requirements. Instead, we observe a situation in which the vehicle continues smoothly in a forward direction while steering back into the path.

From Figure 10 we can see the vehicle better adheres to our bioinspired model of the predator-prey interaction. Instead of reversing direction or undergoing unreasonably tight turns, the vehicle preserves speed and corrects heading to the best of its ability. In the same way, this behavior is largely what we would expect from a cheetah chasing its prey.

Further examination of the behavior is given in Figure 11. Clear limitations in the velocity are observed between 1 m/s and 10 m/s. Additionally, the implemented algorithm provides a smooth velocity command, as expected. This behavior is a result of a continuous mapping of saturation commands back to a region on or within the envelope.

Looking further at Figure 11, we see that the angular rate commands are significantly different from those obtained without saturation constraints. In general, the commanded angular rates achieve higher magnitudes than previously observed. From the third graph, however, it is obvious that there are clear bounds on the ratio of $\omega(t)/v(t)$. As such, the vehicle achieves higher angular rates, while still following the criteria of an Ackermann based platform.

While the saturation constraints perform as desired, it is important to ensure that the overall distance following behavior is not sacrificed as this is a significant trait of our bioinspired desired behavior. Figure 12 demonstrates the comparison of the new following distance to the original. As expected, the added filter (34) between $d^*(t)$ and $d_{\text{ref}}(t)$ introduces a phase shift and attenuation of $d(t)$ when saturation constraints are introduced. Of some interest is the fact that the distance appears more irregular, as opposed to

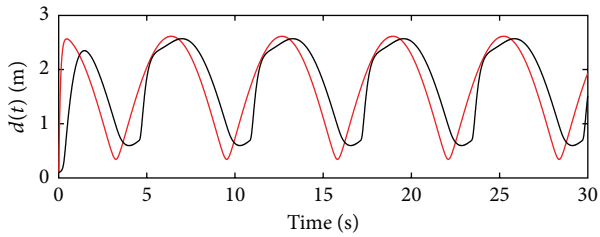

Figure 12: Comparison of following distance with (black) and without (red) saturation.

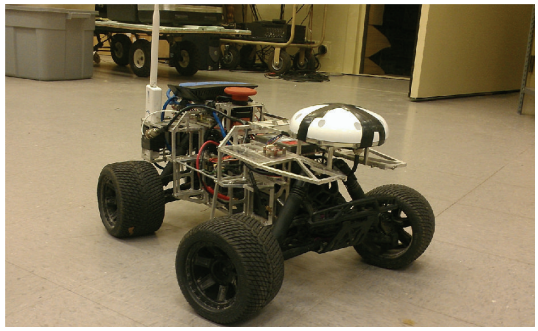

Figure 13: Experimental testing platform.

the smooth original distance. This trend is a result of the fluctuating reference velocity during periods of saturation. Despite these differences, the desired trend is still present in regard to reference system behavior. Consequently, the saturation constraints are held as a valid addition to the overall control algorithm.

5. Experimental Testing and Results

To investigate the practicality of the controller, the algorithm was implemented on an experimental platform. This section discusses the experimental setup, the reference system manipulation, and the results.

5.1. Experimental Setup. For the purpose of testing, the robotic platform is shown in Figure 13. This vehicle incorporates an Inertial Measurement Unit (IMU) that provides Euler angles at $\pm 2^{\circ}$, acceleration on 3 axes at ± 0.01 g, and angular rates at $\pm 0.2^{\circ}$/sec. Additionally, the vehicle hosts an onboard GPS which provides position feedback information within ± 1.5 m as well as velocity information accurate to within ± 0.5 m/s. The sensor information was incorporated into an Extended Kalman Filter, designed specifically for this project, which provided feedback information on position, heading, and velocity, as well as the biases for heading and acceleration measurements.

The onboard actuators consist of a brushless motor for speed and standard hobby servo for steering, both of which were controlled using a generic servo controller. In order to implement the controller in the form presented by this paper, low-level PI controllers were experimentally designed and implemented to directly regulate speed and angular velocity. These controllers, the Kalman Filter, and the overall control algorithm were implemented through an onboard computer at an update rate of 40 Hz.

5.2. Potential Field Reference System Design. For the purpose of experimental testing, the reference system was manipulated according to a simple potential field algorithm similar to that presented in [23–25]. This method was chosen primarily because it offers a computationally simple manner to control the reference system. It also provides an intuitive velocity behavior, which is characterized by reducing velocity when obstacles are encountered.

To define the potential field algorithm, we begin by introducing distance between the reference system and the destination,

$$d_{\text{des}}(t) \triangleq \sqrt{\left(x_{\text{des},i} - x_{\text{r}}(t)\right)^2 + \left(y_{\text{des},i} - y_{\text{r}}(t)\right)^2}, \tag{36}$$

where $d_{\text{des}}(t) \in \mathbb{R}$, $x_{\text{des},i} \in \mathbb{R}$ is the x coordinate of the ith destination waypoint, and $y_{\text{des},i} \in \mathbb{R}$ is the y coordinate of the ith destination waypoint. Furthermore, the distance between the reference system and surrounding obstacles is defined as

$$d_{\text{obs},j}(t) \triangleq \sqrt{\left(x_{\text{obs},j} - x_{\text{r}}(t)\right)^2 + \left(y_{\text{obs},j} - y_{\text{r}}(t)\right)^2}, \tag{37}$$

where $d_{\text{obs},j}(t) \in \mathbb{R}$, $x_{\text{obs},j} \in \mathbb{R}$ is the x coordinate of the jth obstacle waypoint, and $y_{\text{obs},j} \in \mathbb{R}$ is the y coordinate of the jth obstacle waypoint. Based on these distance parameters, virtual force vectors are given such that

$$F_{\text{a}}(t) \triangleq \frac{F_{\text{ac}}}{d_{\text{des}}(t)} \begin{bmatrix} x_{\text{des},i} - x_{\text{r}}(t) \\ y_{\text{des},i} - y_{\text{r}}(t) \end{bmatrix}, \tag{38}$$

$$F_{\text{r}}(t) \triangleq \sum_{j=1}^{N_{\text{obs}}} -F_{\text{rc}} \left(\frac{W}{d_{\text{obs},j}(t)}\right)^n \begin{bmatrix} x_{\text{obs},j} - x_{\text{r}}(t) \\ y_{\text{obs},j} - y_{\text{r}}(t) \end{bmatrix}, \tag{39}$$

where $F_{\text{ac}} > 0$, $F_{\text{rc}} > 0$, $W > 0$, and $n > 1$ are tuning constants, $F_{\text{a}}(t) \in \mathbb{R}^2$ is the attractive force vector, $F_{\text{r}}(t) \in \mathbb{R}^2$ is the repulsive force vector, and N_{obs} is the number of obstacles within a given radius of the reference system.

From (38), we can see that the attractive force, $F_{\text{a}}(t)$, is essentially a unit vector multiplied by the tuning constant F_{ac}. This vector points from the reference system toward the current destination at all times. Because of this relationship, the reference system will be pulled toward the goal with some constant force dictated by F_{ac}.

Likewise, we note that (39) has a similar form to (38). This results in a force vector, $F_{\text{r}}(t)$, that points from a particular obstacle toward the reference system, repelling it. Because of the exponential relationship caused by n, the repulsive force increases as the reference system nears the obstacle. Consequently, the reference system is largely uninfluenced until it is within the vicinity of an obstacle, at which point it is pushed away. Ultimately, the combination of the attractive and repulsive forces, $F_{\text{a}}(t)$ and $F_{\text{r}}(t)$, respectively, causes the reference system to avoid any obstacles while moving toward the goal.

Table 3: Potential field tuning constants for experiments.

Potential Field Parameters
$F_{ac} = 10$
$F_{rc} = 8$
$W = 2$
$n = 1.5$
$E = 14$

Table 4: Common controller tuning parameters for experiments.

Tuning Parameters
$k_v = 1$
$\alpha = 1.3$
$\beta = 1$
$\lambda = 1$
$s = 0.01$
$\omega_d = 2.5$
$\zeta_d = 0.85$
$L = 0.3556$
$\phi_{max} = 25^\circ$
$v_{max} = 10$
$v_{min} = 1$

In order to utilize the force vectors given in (38) and (39), the reference system is given the following dynamics,

$$m\ddot{p}_r(t) + c\dot{p}_r(t) = F_a(t) + F_r(t), \quad (40)$$

where $m > 0$ and $c > 0$ are tuning constants for inertia and damping of the virtual system, respectively. To define these constants, we start by considering the maximum desired velocity in the presence of only attractive forces. When maximum velocity of the reference system is achieved, the reference acceleration reduces to $\ddot{p}_r(t) = 0$, which yields

$$c\dot{p}_r(t) = F_a(t). \quad (41)$$

By taking the norm of both sides and considering the maximum reference velocity case, we can determine the damping as

$$c = \frac{F_{ac}}{v_{r_{max}}}. \quad (42)$$

In regard to determining the inertia, we assume that the reference system persists at maximum velocity throughout the trajectory. If the kinetic energy is maintained along the path, then the path maintains the same characteristic shape, regardless of velocity. As such, the inertial term is defined as

$$m = \frac{2E}{v_{r_{max}}^2}, \quad (43)$$

where $E > 0$ is the user defined kinetic energy term of the reference system. For all experimental tests, the potential field tuning constants used are listed in Table 3. We consider only cases with one or no obstacle presented. Furthermore, only obstacles within a 10 m radius are considered during force computation. For the testing results presented in this paper, $v_{r_{max}}$ is varied accordingly to achieve the desired overall speeds. We do not consider the effect of the potential fields beyond the reference system as the controller is designed to drive the vehicle to the reference system, given that the reference system is composed of class C^1 functions. To better understand the effects of potential fields, we refer the readers to [23–25], which better explore their behavior and usage.

5.3. Results. Each experiment uses the tuning constants laid out in Table 4. In addition, the parameter k_ω is varied to account for actuator dynamics in the steering. Moreover, $v_{r_{max}}$ is altered to achieve varying speeds for each experiment.

We begin by examining the basic tracking performance with no obstacles, presented below in Section 5.3.1. In the three scenarios that follow, the reference velocity is set to constant values of 2 m/s, 5 m/s, and 9 m/s. Each scenario uses the same set of waypoints, which the reference system follows based on the potential field setup presented earlier. Once the reference system is within a meter of the target waypoint, it switches its desired position to the subsequent point. In each of the three cases, the vehicle travels in a clockwise motion. The vehicle starts from rest in each case. We do not consider the exact coordinates of the initial position of the vehicle and reference system. Due to the nature of the experiment, this position is not repeatable and the effects of the initial conditions in these cases are inconsequential relative to the entirety of the path.

Following the tracking performance, we consider applications to obstacle avoidance in Section 5.3.2. Again, we consider three scenarios, a low, a medium, and a high speed case. For these tests, a specific set of waypoints is chosen as well as a single, constant obstacle. The target reference velocity is then varied to examine the behavior of the system. In these tests, data collection starts after the vehicle has reached a velocity relatively close to the intended reference velocity. This decision is to better focus on obstacle avoidance rather than the transient effects of accelerating to the target speed.

5.3.1. Experimental Tracking Performance. For the low speed tracking test, the additional tuning parameters not given in Table 4 are set to $k_\omega = 1$ and $v_{r_{max}} = 2$ m/s. The low speed test results are given in Figure 14. As expected, the vehicle (solid-red) tracks the reference trajectory (dashed-black) tightly. This is the typical behavior at low speeds, as the vehicle lags the reference system by a relatively small amount. While there are some slight deviations at the bottom and top of the graph, the overall performance is favorable in light of experimental results.

In Figure 15, we examine the response of the vehicle to a reference system that is moving at 5 m/s. All parameters are consistent with those in the previous example, with the exception of $k_\omega = 2.5$ and $v_{r_{max}} = 5$ m/s. At first glance, it is clear that the vehicle is cutting to the inside of the reference system's path, as seen in the bottom of the graph and in the top. This performance is ideal based on the derivation of the

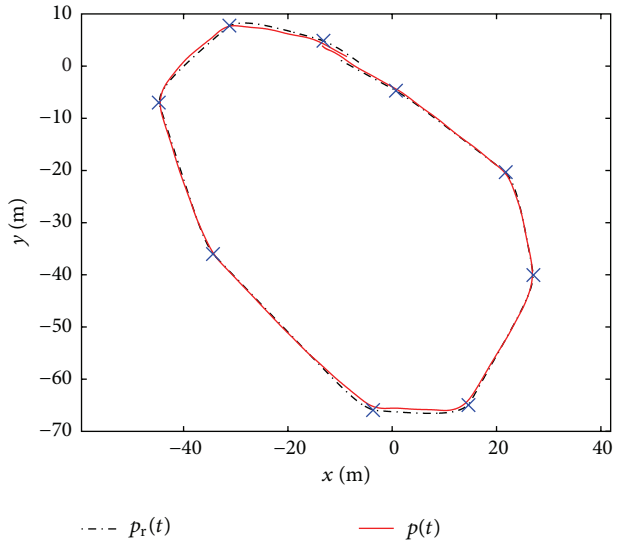

Figure 14: Tracking performance at 2 m/s. The crosses are desired waypoints.

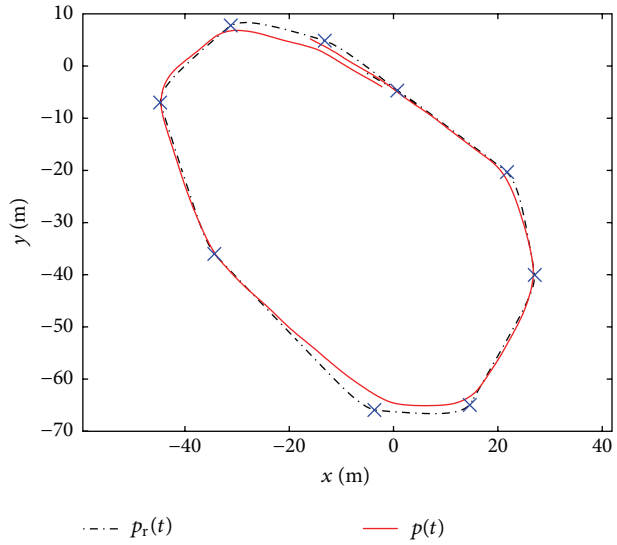

Figure 15: Tracking performance at 5 m/s. The crosses are desired waypoints.

controller. As the vehicle moves faster, the following distance will increase. As a result, the vehicle begins correcting its trajectory earlier. Consequently, the vehicle moves to the inside of the reference system's path, which is corroborated in experimentation.

While the vehicle stays largely within the reference system's path, in the upper left corner of both Figures 14 and 15 the vehicle travels outside the bound defined by the reference trajectory. These mild deviations are attributed to actuation dynamics. In derivation and simulation, we assume that the actuation of longitudinal and angular rates is instantaneous. In reality, there are dynamics that are crucial to the behavior of the system. When the controller commands a tight turn that is within the realm of reachable commands, the dynamics may still keep the vehicle from responding within the necessary time frame.

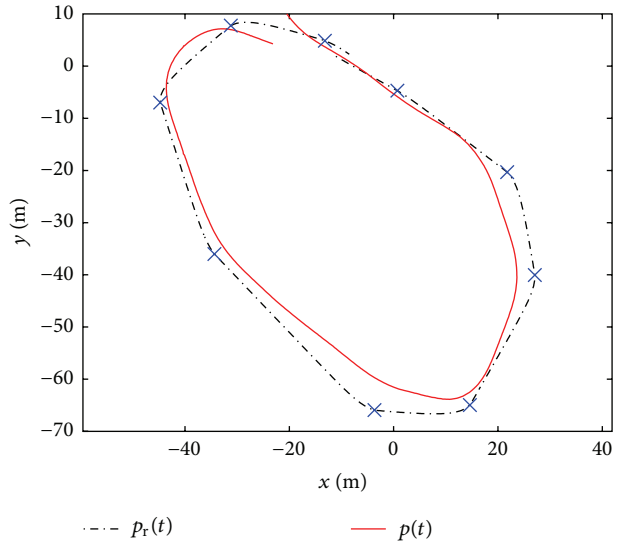

Figure 16: Tracking performance at 9 m/s. The crosses are desired waypoints.

In practice, these dynamic effects can be mitigated by adjusting the tuning parameters k_v and k_ω. By increasing k_ω for the medium speed case, the deviation is substantially less than if the gains would be kept constant from the low speed case.

We refrain from holding k_ω at high values for all tests because the dynamics are also influential at low speeds. With high values of k_ω at low speeds, significant oscillation is seen in the commands, although tracking is largely unchanged.

Next, we examine the tracking performance at high speeds, the results of which are given in Figure 16. Again, all parameters are held constant according to Table 4, with the exception of $k_\omega = 3.5$ and $v_{r_{max}} = 9$ m/s. This test appears as a more extreme case of the medium speed result. In fact, the vehicle's path remains well within the bound laid out by the reference system aside from the initial straightaway and the last turn. As with the medium speed case, the deviation on the last turn is attributed to actuator dynamics, which are not fully compensated for but mitigated by higher values of k_ω.

5.3.2. Application to Obstacle Avoidance. The control algorithm presented can be conceptualized as a position filter, which smoothly follows a reference system despite sudden changes in direction. This behavior elicits the predator-prey interaction sought after. Potential fields, on the other hand, have natural obstacle avoidance properties, but they lack smoothness, which is particularly necessary for high speed situations. Using a combination of the two, we examine the performance of the given approach in the hope of finding more intuitive behavior.

Figures 17 and 18 examine the case of obstacle avoidance at low speed using the same parameters laid out in Table 4, with $k_\omega = 1$ and $v_{r_{max}} = 2$ m/s. In Figures 17, 19, and 21, the triangles and circles denote the location of the reference

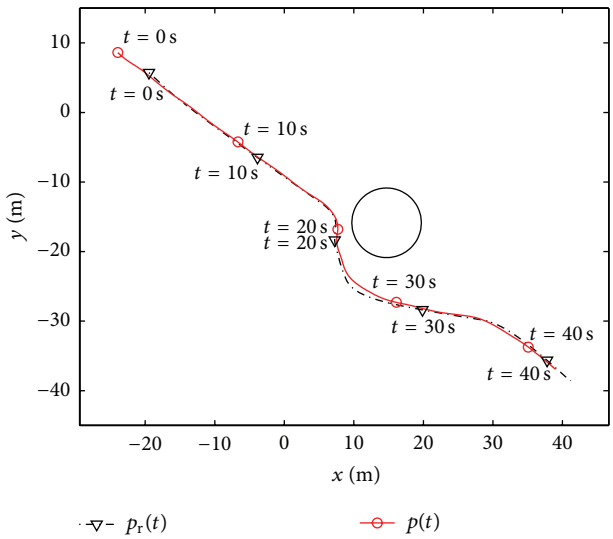

FIGURE 17: Obstacle avoidance at low speeds. Triangles and circles denote the location of reference system and vehicle, respectively, at certain instances in time.

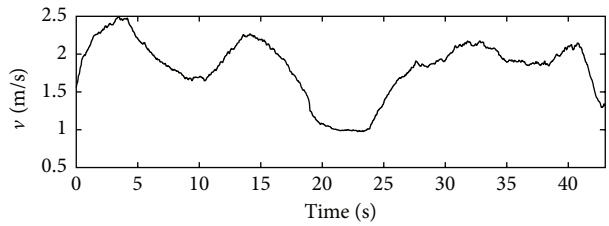

FIGURE 18: Velocity of obstacle avoidance at low speeds.

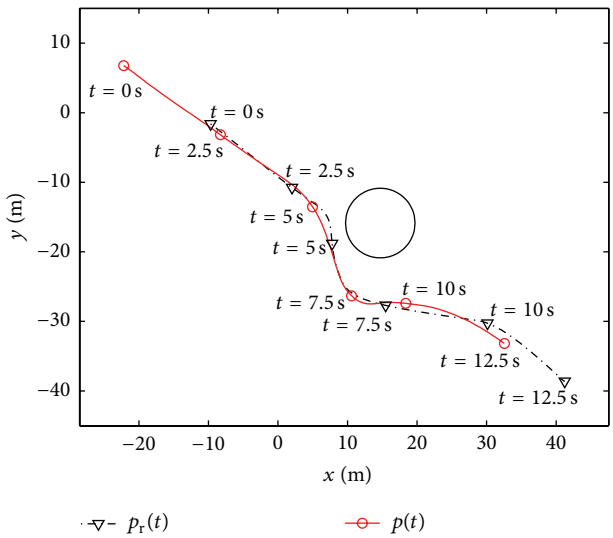

FIGURE 19: Obstacle avoidance at medium speeds. Triangles and circles denote the location of reference system and vehicle, respectively, at certain instances in time.

system and vehicle, respectively, at certain instances in time. Looking at Figure 17, the vehicle tightly tracks the reference

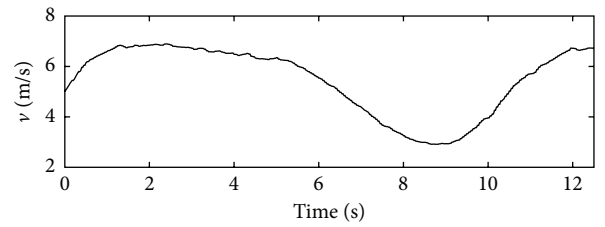

FIGURE 20: Velocity of obstacle avoidance at medium speeds.

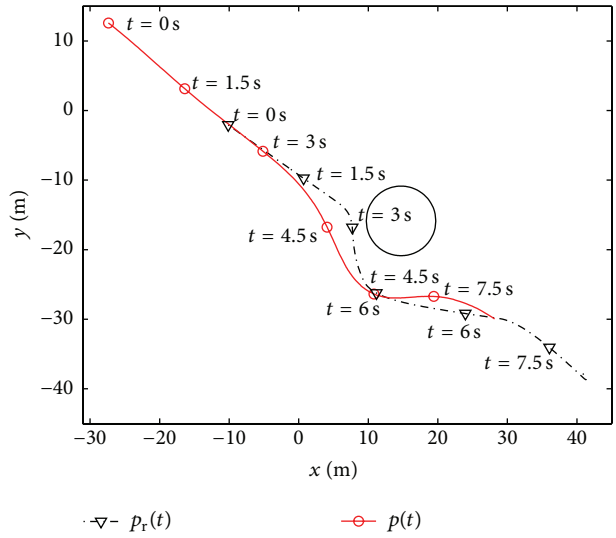

FIGURE 21: Obstacle avoidance at high speeds. Triangles and circles denote the location of reference system and vehicle, respectively, at certain instances in time.

system despite a sharp, awkward corner around the obstacle. Since the speed is small, seen in Figure 18, this behavior is acceptable. Of particular interest is the fact that the velocity drops to only 1 m/s as the system avoids the obstacle. This is a natural result of the potential field system, which we utilize to obtain an intuitive response from the platform. Additionally, we note that this lower velocity is also bounded by the saturation algorithm which maintains at least a 1 m/s speed. Consequently, by examining the time stamps of Figure 17, we see that the distance between the reference and vehicle reduces during the cornering, allowing the vehicle to better track the turn.

The results of obstacle avoidance at medium speeds are given in Figures 19 and 20. Here the tuning gains are altered such that $k_\omega = 2.5$ and $v_{r_{max}} = 6$ m/s. As with the medium speed case in the tracking study, the vehicle does not follow the reference system as tightly. This behavior, however, has distinct advantages in dealing with obstacles. At the 5-second mark, the vehicle begins to diverge from the reference path in order to avoid the obstacle, as opposed to waiting until the last instant. By steering earlier, the vehicle achieves a less severe curve during avoidance, facilitating higher speeds throughout. With regard to the predator-prey analogy, this behavior is more intuitive than basic potential fields. Rather than directly mimicking the prey's path, the predator adjusts

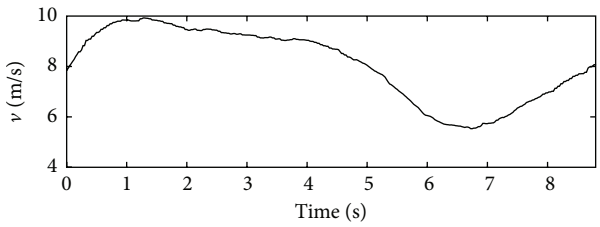

Figure 22: Velocity of obstacle avoidance at high speeds.

its path to easily close in on the prey. The intuition naturally follows for humans as well, as a human driver would avoid immediately upon the observation of an obstacle, rather than waiting. The human's path of avoidance would likewise be based on speed. Looking at Figure 20, the speed decreases from 6 m/s to 3 m/s during the turn. Again, this is a natural behavior as the average driver would naturally slow down in order to safely avoid the obstacle. By slowing down, the distance decreases, which then facilitates better tracking of the reference system.

Lastly, we consider the case of obstacle avoidance at high speeds, seen in Figures 21 and 22. For this case, the tuning gains are altered such that $k_\omega = 2.5$ and $v_{r_{max}} = 9$ m/s. Looking at Figure 21, the algorithm clearly smooths the trajectory of the reference system. In the same way a predator would make milder turns at high speeds, the vehicle departs from the reference path at approximately 3 seconds, at a point much earlier than the reference system begins to avoid. During this outward swerve, Figure 22 indicates that the velocity begins to drop off at 5 seconds. This drop results in a decrease from 9 m/s to 6 m/s, a 33% dip opposed to the 50% change in the previous scenario. With a smoother trajectory, the algorithm allows for more preservation of speed than previously seen. Lastly, we note that the distance between the reference system and vehicle decreases, allowing for a more aggressive turn after passing the obstacle.

6. Conclusions

Many control algorithms currently in existence are largely effective but lack the simplicity of intuitive behavior native to most people. With the increasing push for driverless technologies, however, it is becoming more important to develop algorithms that exploit the trends expected by everyday users in high speed situations. To this end, we consider nature's predators for inspiration. Specifically, our bioinspired model references the cheetah on several occasions. We note that a variable distance to its prey allows it to better plan its path while maintaining a high speed. Additionally, we assume natural constraints on its motion that also hold favorable characteristics for fast maneuvering.

In an effort to reproduce this behavior, a control algorithm, based on a unicycle model, is introduced. This algorithm focuses on driving the current vehicle to within some variable following distance of a user defined reference system by commanding longitudinal and angular rates. This algorithm is proven stable through Lyapunov criteria.

Due to the limitations of the unicycle model, we are not able to directly achieve the sought after motion constraints of the predator. As a result, additional saturation constraints are implemented to enforce Ackermann steering kinematics on the system's behavior. The saturation constraints are presented as a two stage algorithm. The first stage, which is not a sufficient method within itself, focuses on preventing chatter in the final command. The second stage receives commands from the first stage and maps them within an achievable envelope. From the saturation methodologies, the reference system is then manipulated to satisfy the Lyapunov criteria. Based on simulations, the dual stage saturation algorithm exhibits the desired intuitive behavior.

Using potential fields to control the reference system, the controller is implemented on a small scale robotic vehicle. Testing results thus far indicate that the proposed controller, coupled with a potential field algorithm, exhibits strong intuitive behaviors similar to those described by the predator-prey relationship. At low speeds, the vehicle tracks the reference system within a close tolerance of the reference system's path. This trend holds in the case of obstacle avoidance. However, at high speeds, the vehicle diverges from the set path and instead smooths the overall trajectory, allowing some conservation of speed without extremely aggressive steering, much as would be expected from a human controller. Overall, the algorithm was used to successfully navigate trajectories at up to 9 m/s.

There are several instances in which the dynamics prevent the vehicle from consistently following the idealized path. Consequently, future work will focus on including these effects through backstepping or other related methods. Also, the algorithm will be refined for applicability to multiple platforms, most likely through adaptive methods.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This material is based upon the work performed while Dr. Alexander Leonessa was serving at the National Science Foundation of the United States. The authors would also like to thank Virginia Tech's Open Access Subvention Fund for their support in publishing this work.

References

[1] K. Chu, M. Lee, and M. Sunwoo, "Local path planning for off-road autonomous driving with avoidance of static obstacles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1599–1616, 2012.

[2] S. Moon, U. Lee, and D. H. Shim, "Study on real-time obstacle avoidance for unmanned ground vehicles," in *Proceedings of the International Conference on Control, Automation and Systems (ICCAS '10)*, pp. 1332–1335, October 2010.

[3] D. Fu-guang, J. Peng, B. Xin-qian, and W. Hong-jian, "AUV local path planning based on virtual potential field," in *Proceedings*

of the IEEE International Conference on Mechatronics and Automation (ICMA '05), vol. 4, pp. 1711–1716, August 2005.

[4] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," in *Proceedings of the American Control Conference*, pp. 4609–4614, June 2008.

[5] P. Zhao, J. Chen, T. Mei, and H. Liang, "Dynamic motion planning for autonomous vehicle in unknown environments," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV '11)*, pp. 284–289, Baden-Baden, Germany, June 2011.

[6] S. Glaser, B. Vanholme, S. Mammar, D. Gruyer, and L. Nouvelière, "Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 589–606, 2010.

[7] M. Häselich, N. Handzhiyski, C. Winkens, and D. Paulus, "Spline templates for fast path planning in unstructured environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '11)*, pp. 3545–3550, IEEE, San Francisco, Calif, USA, September 2011.

[8] C. Fulgenzi, A. Spalanzani, and C. Laugier, "Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 1610–1616, April 2007.

[9] A. Richardson and E. Olson, "Iterative path optimization for practical robot planning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '11)*, pp. 3881–3886, September 2011.

[10] P. Bhattacharya and M. L. Gavrilova, "Voronoi diagram in optimal path planning," in *Proceedings of the 4th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD '07)*, pp. 38–47, July 2007.

[11] K. P. Valavanis, T. Hebert, R. Kolluru, and N. Tsourveloudis, "Mobile robot navigation in 2-D dynamic environments using an electrostatic potential field," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, vol. 30, no. 2, pp. 187–196, 2000.

[12] M. Park and M. Lee, "Experimental evaluation of robot path planning by artificial potential field approach with simulated annealing," in *Proceedings of the 41st SICE Annual Conferenceerence*, vol. 4, pp. 2190–2195, Osaka, Japan, August 2002.

[13] X. Hu, "Clonal selection based mobile robot path planning," in *Proceedings of the IEEE International Conference on Automation and Logistics (ICAL '08)*, pp. 437–442, September 2008.

[14] A. M. Wilson, J. C. Lowe, K. Roskilly, P. E. Hudson, K. A. Golabek, and J. W. Mc-Nutt, "Locomotion dynamics of hunting in wild cheetahs," *Nature*, vol. 498, no. 7453, pp. 185–189, 2013.

[15] R. Fierro and F. L. Lewis, "Control of a nonholonomic mobile robot: backstepping kinematics into dynamics," *Journal of Robotic Systems*, vol. 14, no. 3, pp. 149–163, 1997.

[16] D. DeVon and T. Bretl, "Kinematic and dynamic control of a wheeled mobile robot," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 4065–4070, San Diego, Calif, USA, November 2007.

[17] K. Pathak and S. K. Agrawal, "Planning and control of a nonholonomic unicycle using ring shaped local potential fields," in *Proceedings of the American Control Conference (AAC '04)*, pp. 2368–2373, July 2004.

[18] Q. Wang and Y.-P. Tian, "Formation control of multiple nonholonomic unicycles using adaptive perturbation method," in *Proceedings of the 51st IEEE Conference on Decision and Control (CDC '12)*, pp. 2453–2458, December 2012.

[19] L. C. Figueiredo and F. G. Jota, "Nonholonomic systems stabilization and tracking control using discontinuous control," in *Proceedings of the IEEE Conference on Automation Science and Engineering (CASE '05)*, pp. 142–147, August 2005.

[20] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle like vehicles via Lyapunov techniques," *IEEE Robotics & Automation Magazine*, vol. 2, no. 1, pp. 27–35, 1995.

[21] R. Carona, A. P. Aguiar, and J. Gaspar, "Control of unicycle type robots," in *IV Jornadas de Engenharia Electronica e Telecomunicacoes e de Computadores*, pp. 180–185, November 2008.

[22] V. Medina-Garciadiego and A. Leonessa, "Tracking control of a nonholonomic ground vehicle," in *Proceedings of the American Control Conference (ACC '11)*, pp. 1710–1713, IEEE, San Francisco, Calif, USA, June-July 2011.

[23] J. Borenstein and Y. Koren, "High-speed obstacle avoidance for mobile robots," in *Proceedings of the 3rd International Symposium on Intelligent Control*, pp. 382–384, August 1988.

[24] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989.

[25] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 1398–1404, April 1991.