A New Analytical Solution to Mobile Robot Trajectory Generation in the Presence of Moving Obstacles

Zhihua Qu, *Senior Member, IEEE*, Jing Wang, *Member, IEEE*, and Clinton E. Plaisted, *Member, IEEE*

***Abstract*—The problem of determining a collision-free path for a mobile robot moving in a dynamically changing environment is addressed in this paper. By explicitly considering a kinematic model of the robot, the family of feasible trajectories and their corresponding steering controls are derived in a closed form and are expressed in terms of one adjustable parameter for the purpose of collision avoidance. Then, a new collision-avoidance condition is developed for the dynamically changing environment, which consists of a time criterion and a geometrical criterion, and it has explicit physical meanings in both the transformed space and the original working space. By imposing the avoidance condition, one can determine one (or a class of) collision-free path(s) in a closed form. Such a path meets all boundary conditions, is twice differentiable, and can be updated in real time once a change in the environment is detected. The solvability condition of the problem is explicitly found, and simulations show that the proposed method is effective.**

***Index Terms*—Car-like robot, chained form, moving obstacle, nonholonomic systems, obstacle avoidance, piecewise parameterization, polynomial inputs, trajectory generation.**

I. Introduction

For most real-world applications, it is desirable that mobile robots are capable of exploring or moving within a dynamic environment. In addition, the environment is usually uncertain as complete information and future trajectories of obstacles cannot be assumed *a priori*. In this context, the problem naturally arising is how to plan in real time a collision-free path in the presence of dynamically moving objects and with a limited sensing range. A preferred solution to the problem would be one that takes kinematic constraints into consideration, explicitly handles dynamically moving objects, and is analytical.

Standard motion-planning approaches [1], such as potential field [2] and vector field histogram [3], are developed to deal with geometrical constraints, more specifically, holonomic systems in the presence of static obstacles. For nonholonomic systems such as mobile robots, their kinematic constraints make time derivatives of some configuration variables nonintegrable, and, hence, a collision-free path in the configuration space is not necessarily feasible (that is, it may not be achievable by steering controls) [4], [5]. Up to now, most of the existing results deal with nonholonomic systems and object avoidance in one of two ways. One way is to exclusively focus upon motion planning under nonholonomic constraints. Without considering obstacles, many algorithms have been proposed, for instance, differential geometry [6], differential flatness [7], input parameterization [8]–[10], and optimal control [11]. In particular, the nonholonomic motion-planning problem can be recast as an optimal control problem, Pontryagin's Maximum Principle can be applied, and it is shown in [12] (improved later in [13]) that the feasible shortest path for a point robot under two boundary conditions is a concatenation of simple pieces (such as an arc and a straight line segment) that belong to 46 three-parameter families of controls. The second way is to modify the result from a holonomic planner so the resulting path is feasible. For example, the online suboptimal obstacle avoidance algorithm in [14] is based on the Hamilton–Jacobi–Bellman equation [15], [16], it admits stationary obstacles, a planned path is holonomic, and its feasibility has to be verified for a chosen nonholonomic mobile robot. The nonholonomic path planner in [17] is based on the same principle, that is, a path is generated by ignoring nonholonomic constraints, and it is then made feasible via approximation by using a sequence of such optimal path segments as those in [12].

Exhaustive search or numerical iteration-based methods have also been used to deal with nonholonomic constraints and collision avoidance. The search-based algorithm [18] involves discretization of the configuration space in order to build and search a graph whose nodes are small axis-parallel cells; two cells are called to be adjacent if there is a feasible path segment between them, and these path segments are constructed by discretizing the controls and integrating the equations of motion. In [19], nonholonomic motion planning is formulated as a nonlinear least-squares problem in an augmented space, obstacle avoidance is included as inequality constraints, and a solution is found numerically. In [20], a dynamic programming-based algorithm is introduced to search approximately for minimum-time trajectories and to take into account both kinematic constraints of avoiding static obstacles and dynamic constraints in terms of bounds on velocity, acceleration, and force. In [21], trajectory planning (so-called kinodynamic planning) is pursued by considering first-order differential equations and static obstacles and by finding appropriate inputs through a random tree search.

Manuscript received April 24, 2003; revised November 25, 2003. This paper was recommended by Associate Editor N. Amato and Editor A. De Luca upon evaluation of the reviewers' comments. This work was supported in part by Lockheed Martin Corporation. This paper was presented at the Florida Conference on Recent Advances in Robotics, Boca Raton, FL, May 8–9, 2003.

Z. Qu and J. Wang are with the Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816 USA (e-mail: qu@mail.ucf.edu).

C. E. Plaisted was with Lockheed Martin Missiles and Fire Control, Orlando, FL 32819 USA. He is now with NASA Expendable Launch Vehicles, Mission Analysis Branch, A.I. Solutions, Inc., Kennedy Space Center, Orlando, FL 32899 USA.

Digital Object Identifier 10.1109/TRO.2004.829461

There have been a few results on dealing with moving obstacles. It is proposed in [22] that, if the entire trajectories of the moving obstacles are known *a priori*, an $(n+1)$-dimensional configuration-time space can be formed by treating the time as a state variable and recasting the dynamic motion-planning problem into a static one. In [23], kinodynamic motion planning with moving obstacles is done using a randomized motion planner in which a control is chosen randomly from the set of admissible values to integrate equation of motion and, if the resulting local trajectory is collision-free, its endpoint is put into a probabilistic roadmap. Similar to the approach in [18], the random motion planning of [23] is a search method. In [24], the dynamic motion-planning problem is decomposed into two subproblems: a static path-planning problem and a velocity-planning problem. The static path-planning problem is to find a path that avoids all static obstacles, and the velocity-planning problem is to determine the velocity of the robot along that path so that there will be no collision. However, this approach requires complete information (including future trajectories), and its solution is not guaranteed. For obstacles moving with known constant velocities, velocity planning can be done using the velocity obstacle concept in [25]. That is, collision does not occur if the robot velocity is chosen such that its velocities relative to the obstacles' motion do not enter the corresponding collision cones. To the best of our knowledge, there has been no comprehensive result on analytical motion planning for nonholonomic systems operating in a dynamical and uncertain environment.

In this paper, a new collision-avoidance method is proposed to analytically solve the problem of real-time trajectory planning and replanning for nonholonomic mobile robots operating in a environment of multiple dynamically moving obstacles. The proposed method is a three-leg paradigm: mapping the system kinematic constraints into a chained form, parameterizing all feasible trajectories by a family of piecewise-constant polynomials and determining the corresponding steering controls, and developing a new collision-avoidance condition (which consists of a time criterion and a geometrical criterion and is explicit in both the transformed space and the original working space). Specifically, it has been shown that, for a car-like mobile robot (and others in the (2, 4) chained form), a family of sixth-order piecewise-constant polynomials can be used to describe feasible trajectories (for which steering controls are explicitly found) and that, upon satisfying all boundary conditions, collision-free trajectories can be expressed in terms of one parameter. This parameterization makes it possible to analytically solve for collision-free path(s) by invoking the proposed collision-avoidance condition. The resulting trajectory is twice differentiable, and the corresponding steering controls are piecewise continuous. As a result of the piecewise representations used, the paradigm works if obstacles have varying speeds and if on-board sensors have a limited range. Iteration (i.e., updating the planned trajectory) is only needed when speed changes of objects are detected or when new objects emerge, and the iteration itself is done in a closed form and limited to one set of calculations (that is, no successive substitution or iteration). It is shown that, so long as collision does not occur at the boundary conditions and there is no perpetual disconnectivity (such as a lasting trap), the dynamic path generation problem is always solvable and that solutions are given in closed form.

This paper is organized into five sections. In Section II, the problem of real-time motion planning in the presence of dynamically moving obstacles is formulated, a four-wheel mobile robot is used as the plant, and its chained form is presented. In Section III, a piecewise-constant parameterization of feasible trajectories is introduced, a new collision-avoidance condition (in both time and geometry) is developed, and analytic solutions to collision-free trajectory and steering functions are obtained. Simulation results on path planning, replanning, and steering of the car-like mobile robot are discussed Section IV. In Section V, several conclusions are drawn.

II. Problem Formulation

In this paper, we shall consider the general problem of trajectory planning for mobile robots in a dynamic and changing environment. As shown in Fig. 1, possible two-dimensional (2-D) environmental changes are due to limited ranges of on-board sensors and to the appearance of and/or motion of objects. To solve the problem, one can make the following choices without loss of any generality:[1]

- The robot under consideration is represented by a 2-D circle with the center at $O(t) = (x, y)$ and of radius R. Its motion is controlled but *nonholonomic* and is represented by the velocity vector $v_r(t)$. The range of its sensors is also described by a circle centered at $O(t)$ and of radius R_s.
- The ith object, $i = 1, \ldots, n_o$, will be represented by a circle centered at point $O_i(t)$ and of radius r_i, denoted by $B_i(O_i(t), r_i)$. For moving objects, the origin $O_i(t)$ is time varying and moving with linear velocity vector $v_i(t)$.
- The robot starts at initial position O_o and initial orientation θ_0, moves collision-free, and arrives at final position O_f and with final orientation θ_f.

Intuitively, the trajectory-planning problem has at least one solution if the robot is capable of moving sufficiently fast, if the free space contains the initial condition at $t = t_0$, and if there exists a finite time instant $T_f > 0$ such that the free space is connected and contains the final position for $t \geq t_0 + T_f$ and for all $i = 1, \ldots, n_o$. The condition of solvability and its implications will be mathematically introduced and then discussed in Section III-C.

However, the general trajectory-planning problem is physically ill-posed as its solution will require *a priori* knowledge of both the objects' present and future motion information. To overcome this difficulty while making the proposed method practically implementable, we use piecewise constants and functions to represent arbitrary functions. Specifically, within a specified period of time $t \in [t_0 + kT_s, t_0 + (k+1)T_s)$ (where T_s is often small):

- velocity v_i of the ith object is constant, denoted by v_i^k;
- only the objects in the range of sensors are considered;

[1]It is trivial to allow the envelope of either the robot or an obstacle to be represented by union/intersection of several circles. The envelopes could also be polygonal. Mathematically, circular envelopes can be represented by second-order inequalities while polygonal envelopes can be described by first-order linear inequalities.

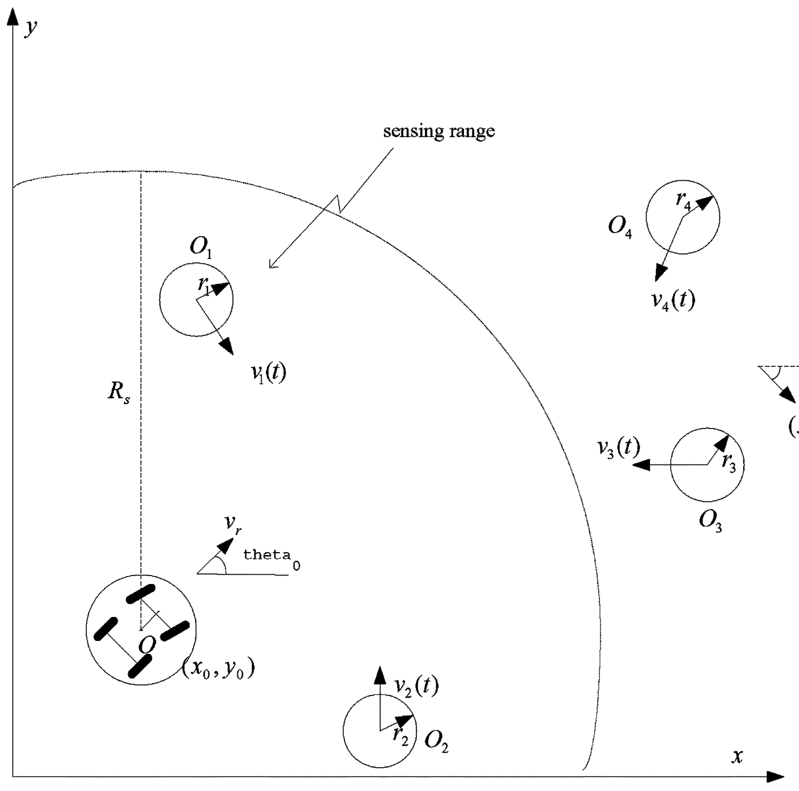

Fig. 1. General setting of trajectory planning in the presence of moving obstacles.

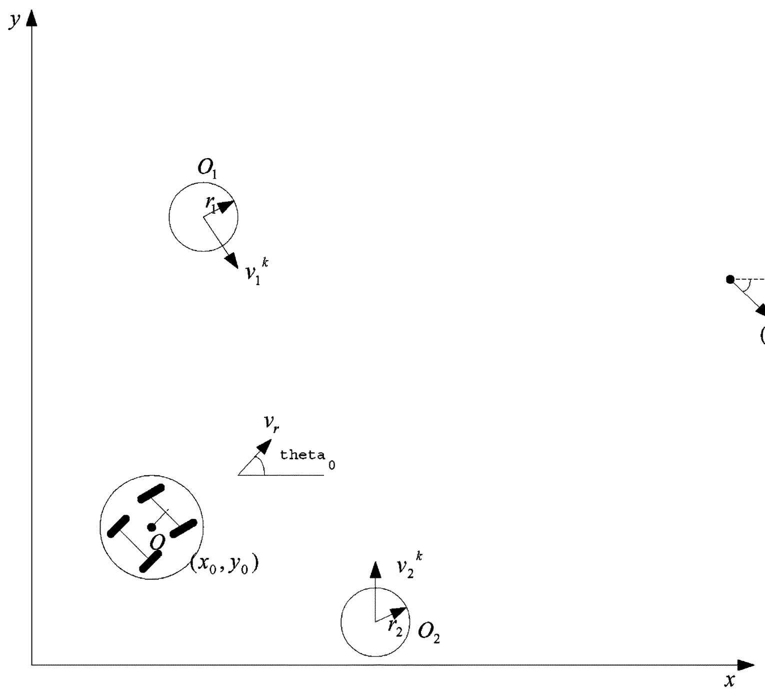

Fig. 2. Simplified setting of trajectory planning in the presence of moving obstacles.

- trajectory and control of the robot are chosen to be functions with piecewise-constant parameters.

In some application, not only is the sensor range limited, the final position O_f may not be fixed either and thus can also be represented by a piecewise-constant function. Therefore, trajectory planning or replanning is done for Fig. 2, a snapshot of Fig. 1, and is constantly updated. To do so efficiently online, the proposed piecewise-constant parameterization must yield analytical solutions.

A. Velocity Cones

The simplification of approximating an arbitrarily moving object by a constant velocity (or linearly moving) object can always be done for a period of time. In the case that only

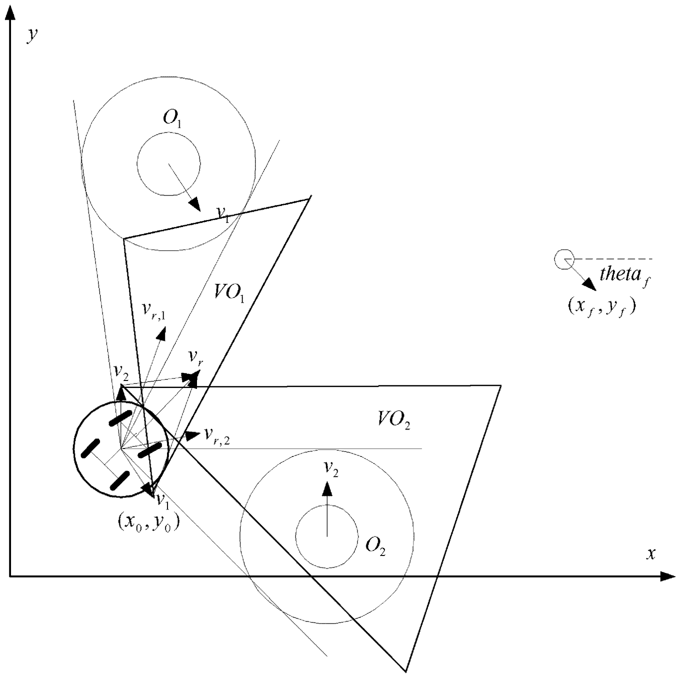

Fig. 3. Trajectory planning in the presence of velocity objects—velocity cone method.

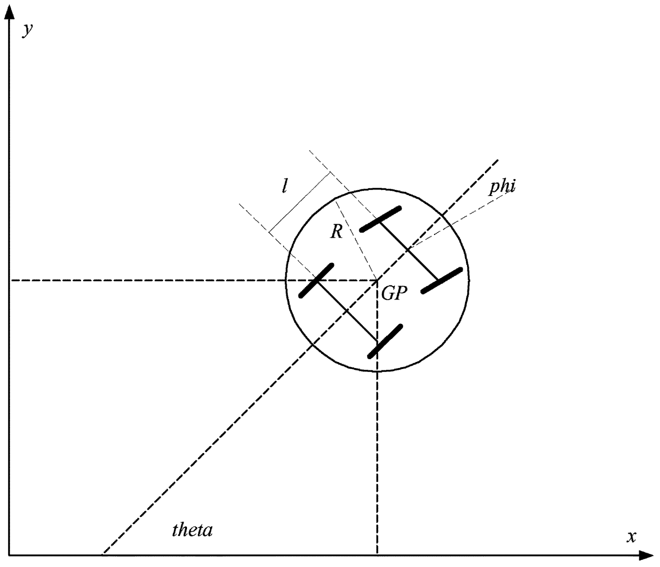

Fig. 4. Car-like robot.

linear velocity objects are present, the method of velocity planning [25] can be used to solve the problem of obstacle avoidance. To illustrate its idea, consider Fig. 3 in which two objects are moving with constant velocities v_1 and v_2 and the objects are enlarged by the radius of the robot. Given an arbitrary robot velocity v_r (of its guidepoint (GP), i.e., $v_r = [\dot{x}, \dot{y}]^T$), the relative velocities $v_{r,i} = v_r - v_i$ can be determined. It is straightforward to verify graphically that, if the relative velocities do not enter the two velocity cones (which are corresponding to and are pointing toward the objects), respectively, collision will never happen. As shown in Fig. 3, $v_{r,1}$ is in the cone of object 1, and $v_{r,2}$ is not in the cone of object 2. As a result, if all three velocities are maintained, the robot will collide with object 1 but not with object 2. Thus, as proposed in [25], velocity planning of v_r should be done so that $v_{r,i}$ are not in their corresponding cones. In [26], the concept of the velocity cone is extended to "nonlinear velocity" objects by taking into account the information of shape, velocity, and path curvature of a moving obstacle along a known trajectory. Nonetheless, the basic idea remains the same, that is, choosing robot velocity so that the relative velocities do not enter the corresponding velocity cones.

While the concept of the velocity cone is simple and intuitive, it has several shortcomings if applied to the problem defined previously in Figs. 1 and 2.

- In most cases, velocities of the robot and objects are not constants and consequently it is not necessary or sufficient for collision avoidance that the relative velocities are not in the corresponding cones.
- The decomposition of trajectory planning into two subproblems of path planning and velocity planning is inadequate for a truly dynamic environment as all obstacles and their complete trajectories must be known *a priori*.
- Since path planning is a kinematic problem and velocity planning is a dynamic problem, kinematic constraints and the dynamic model of the robot must be considered in trajectory planning. Most of the existing work give little consideration to either.

B. Robot Modeling

In this paper, a new paradigm is proposed to plan trajectories and avoid moving obstacles for nonholonomic mobile robots. In the new paradigm, the kinematic models of the robots are explicitly considered in trajectory planning, and the dynamic models could also be included (though the latter will not be considered in this paper). To this end, a kinematic model of a car-like mobile robot is used in this paper to develop a trajectory-planning algorithm using the paradigm.

The car-like robot is shown in Fig. 4, its front wheels are steering wheels, and its rear wheels are driving wheels but have a fixed orientation. The distance between the two wheel-axle centers is l, the midpoint along the line connecting the axle centers is set to be the GP, and the whole vehicle is physically within a circle of radius R and centered at the GP. Trajectory planning will be done for the GP. Let the generalized coordinates be $q = [x\, y\, \theta\, \phi]^T$, where (x, y) are the Cartesian coordinates of the GP, θ is the orientation of the robot body with respect to the x axis (that is, the slope angle of the line passing through the GP and center of the back axle), and ϕ is the steering angle.

Let ρ be the radius of the (back) driving wheels, u_1 be the angular velocity of the driving wheels, and u_2 be the steering rate of the (front) guiding wheels. One can obtain the following kinematic model for the car-like robot:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{pmatrix} = \begin{pmatrix} \rho\cos(\theta) - \frac{\rho}{2}\tan(\phi)\sin(\theta) & 0 \\ \rho\sin(\theta) + \frac{\rho}{2}\tan(\phi)\cos(\theta) & 0 \\ \frac{\rho}{l}\tan(\phi) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}. \quad (1)$$

Kinematic model (1) has singularity at $\phi = \pm\pi/2$, which does not occur mathematically or in practice by limiting the range of ϕ within $(-\pi/2\, \pi/2)$.

In order to standardize the process of analytically solving the trajectory-planning problem for model (1) and for other applicable models as well, we choose to proceed with our development by first transforming kinematic model (1) into a canonical form, called the chained form. Following the standard results on the chained form, one can find following transformations of coordinates and inputs:

$$\begin{aligned} z_1 &= x - \frac{l}{2}\cos(\theta) \\ z_2 &= \frac{\tan(\phi)}{l\cos^3(\theta)} \\ z_3 &= \tan(\theta) \\ z_4 &= y - \frac{l}{2}\sin(\theta) \qquad (2) \\ u_1 &= \frac{v_{c1}}{\rho\cos(\theta)} \\ u_2 &= -\frac{3\sin(\theta)}{l\cos^2(\theta)}\sin^2(\phi)v_{c1} + l\cos^3(\theta)\cos^2(\phi)v_{c2}. \qquad (3) \end{aligned}$$

Under the transformations, kinematic model (1) can be mapped into the following two-input four-state chained form:

$$\begin{aligned} \dot{z}_1 &= v_{c1} \\ \dot{z}_2 &= v_{c2} \\ \dot{z}_3 &= z_2 v_{c1} \\ \dot{z}_4 &= z_3 v_{c1}. \end{aligned} \qquad (4)$$

Remark 2.1: State and control transformations (2) and (3) are one to one if $\theta \neq \pm\pi/2$. This diffeomorphism can always be guaranteed by properly prerotating the x–y plane and making $\theta(t) \in (-\pi/2, \pi/2)$ provided that boundary conditions θ_0 and θ_f on θ satisfy the inequality $|\theta_0 - \theta_f| < \pi$. If $|\theta_0 - \theta_f| \geq \pi$, it becomes necessary to introduce an intermediate configuration with θ_m such that $|\theta_0 - \theta_m| < \pi$ and $|\theta_m - \theta_f| < \pi$. In this case, trajectory planning from θ_0 to θ_f can be done by solving the problem twice, one from θ_0 to θ_m and another from θ_m to θ_f. $\diamond$

Remark 2.2: It should be noted that a collision-avoidance criterion has to be established in the physical space (e.g., the plane of x versus y) rather than the transformed space specified by the chained form. Nonetheless, the chained form provides the standardization by which most of the derivations (such as path parameterization and steering controls) are invariant. For example, if the GP is set at the real-axle midpoint, chained form (4) can be obtained except that $z_1 = x$ and $z_4 = y$ [rather than those in (2)]. In this case, simpler criteria for obstacle avoidance can be derived by following the discussions in Section III-B, but the result will become conservative as radius R has to be increased to cover the vehicle. Similarly, the proposed method can be applied to other types of robotic vehicles. $\diamond$

In this paper, we use the car-like robot as the example and adopt the chained form in solving the problem of trajectory planning. The proposed steering paradigm for trajectory planning and object avoidance has the following features.

- Kinematic models of robots are explicitly considered.
- Motion of objects are represented by piecewise constant velocities, and collision-avoidance criterion is defined analytically and thus less conservative than the existing methods.
- Piecewise-constant parameterization will be used to define trajectory and steering control, and their solutions are obtained in closed form.

III. Proposed Steering Paradigm

The proposed paradigm consists of three basic steps, and it is based on the two corner stones of steering and collision-free criterion (newly defined for moving objects). On one side, it begins with a kinematic model, that is, steering strategies are used to find out the class of physically achievable trajectories. On the other hand, a collision-avoidance criterion can be explicitly developed for moving objects. As the third step, a specific class within all achievable trajectories will first be parameterized and then solved using the object avoidance criterion.

A. Feasible Trajectories

A trajectory is *feasible* if it satisfies both the boundary conditions imposed and dynamics of the kinematic model. The chained form in (4) is used as the standard one to study and determine trajectories that observe the kinematic model. The following result shows a general class of feasible trajectories in terms of transformed state z.

Lemma 1: Consider a mobile robot that has its kinematic model in the chained form (4) and operates in an obstacle-free environment. Then, given any boundary conditions $z(t_0) = z^0 = [z_1^0, z_2^0, z_3^0, z_4^0]^T$ and $z(t_f) = z^f = [z_1^f, z_2^f, z_3^f, z_4^f]^T$ (for some $t_f > t_0$), there exist inputs v_{c1} and v_{c2} that render a feasible trajectory of functional form $z_4 = F(z_1)$ (in the plane of z_1 and z_4). In particular, any function $z_4 = F(z_1)$ is feasible as long as it satisfies the following boundary conditions:

$$\begin{aligned} z_4^0 &= F\left(z_1^0\right) \\ z_4^f &= F\left(z_1^f\right) \\ z_3^0 &= \frac{dF\left(z_1^0\right)}{dz_1^0} \\ z_3^f &= \frac{dF\left(z_1^f\right)}{dz_1^f} \\ z_2^0 &= \frac{d^2F\left(z_1^0\right)}{d\left(z_1^0\right)^2} \\ z_2^f &= \frac{d^2F\left(z_1^f\right)}{d\left(z_1^f\right)^2}. \end{aligned}$$

Proof: For a function of form $z_4 = F(z_1)$ to qualify as a feasible trajectory, all of the boundary conditions must be met, and they are satisfied because, according to (4)

$$\begin{aligned} z_3 &= \frac{dz_4}{dz_1} \\ z_2 &= \frac{d^2z_4}{d(z_1)^2}. \end{aligned}$$

Clearly, there are many choices of $v_{c1}(t)$ to drive $z_1(t)$ from z_1^0 to z_1^f. By choosing $v_{c1}(t)$ to be nonzero, $v_{c2}(t)$ can be found by differentiating $z_4(t) = F(z_1(t))$ according to (4). This concludes the proof. $\diamondsuit$

Remark 3.1: If $z_1^0 \neq z_1^f$, the first control component can be set to be $v_{c1}(t) = C$ for some nonzero constant C. If $z_1^f = z_1^0$, $v_{c1} = 0$ would be the constant control but cause singularity in determining $v_{c2}(t)$. In the latter case, the singularity problem with $v_{c1}(t)$ being zero can be overcome by first choosing an intermediate point z_1^m with $z_1^m \neq z_1^0 = z_1^f$ and then by proceeding with planning two subpaths in the plane of z_1 and z_4, that is, letting $z_1(t)$ move from z_1^0 to z_1^m and then back to $z_1^f = z_1^0$. By doing so, the resulting control $v_{c1}(t)$ will be piecewise constant but nonzero. $\diamondsuit$

Lemma 1 shows that, by first choosing $F(\cdot)$ and then making $z_4 = F(z_1)$ conform the boundary conditions $(x_0, y_0, \theta_0, \phi_0)$ and $(x_f, y_f, \theta_f, \phi_f)$ in the original state space, the steering problem can be solved. In this paper, it is assumed that $\phi_0 = \phi_f = 0$. Thus, for a feasible trajectory, the following boundary conditions on boundary points, slopes, and curvatures are applied: given $z_4 = F(z_1)$, we have

$$\begin{cases} z_1^0 = x_0 - \frac{l}{2}\cos(\theta_0), & F\left(z_1^0\right) = y_0 - \frac{l}{2}\sin(\theta_0) \\ \left.\frac{dz_4}{dz_1}\right|_{z_1=z_1^0} = \tan(\theta_0), & \left.\frac{d^2z_4}{d(z_1)^2}\right|_{z_1=z_1^0} = 0 \end{cases} \tag{5}$$

$$\begin{cases} z_1^f = x_f - \frac{l}{2}\cos(\theta_f), & F\left(z_1^f\right) = y_f - \frac{l}{2}\sin(\theta_f) \\ \left.\frac{dz_4}{dz_1}\right|_{z_1=z_1^f} = \tan(\theta_f), & \left.\frac{d^2z_4}{d(z_1)^2}\right|_{z_1=z_1^f} = 0. \end{cases} \tag{6}$$

Remark 3.2: For the class of feasible trajectories in the form of $z_4 = F(z_1)$, boundary conditions in (5) and (6) represent six constraint equations. Thus, function $z_4 = F(z_1)$ can be chosen to be a polynomial of fifth order or higher. $\diamondsuit$

Remark 3.3: If $\phi_0 = \phi_f = 0$ is not imposed, then the boundary curvatures in boundary conditions (5) and (6) should be changed to

$$\left.\frac{d^2z_4}{dz_1^2}\right|_{z_1=z_1^0} = \frac{\tan(\phi_0)}{l\cos^3(\theta_0)}$$
$$\left.\frac{d^2z_4}{dz_1^2}\right|_{z_1=z_1^f} = \frac{\tan(\phi_f)}{l\cos^3(\theta_f)}. \tag{7}$$

The above boundary conditions on the second-order derivatives, together with those on first-order derivatives, are equivalent to boundary curvatures (κ) of the trajectory as

$$\kappa = \frac{\frac{d^2z_4}{dz_1^2}}{\left[1 + \left(\frac{dz_4}{dz_1}\right)^2\right]^{3/2}}.$$

$\diamondsuit$

In the next subsection, collision avoidance is studied for a feasible trajectory. Later, a specific class of feasible trajectories will be defined to yield a closed-form solution for collision avoidance and steering control.

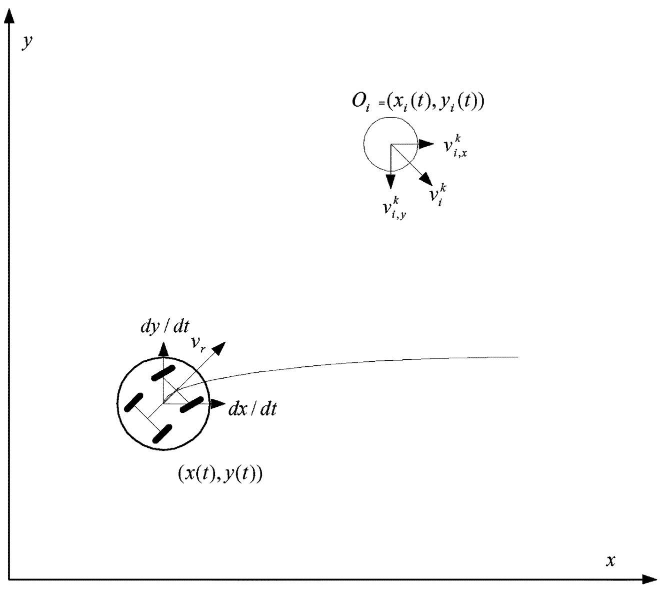

Fig. 5. Steering paradigm: robot and the ith object.

B. Criterion for Avoiding Dynamic Objects

To illustrate the criterion in the proposed steering paradigm, consider the robot (of coordinates $(x(t), y(t))$) and the ith object (of coordinates $(x_i(t), y_i(t))$) in Fig. 5 for the period $t \in [t_0 + kT_s, t_0 + (k+1)T_s)$. In the figure, the robot is moving at a vector velocity $v_r \triangleq [\dot{x}(t)\,\dot{y}(t)]^T$ (which is to be determined), the object has an initial location $O_i = (x_i^k, y_i^k)$ where $x_i^k = x_i(t_0 + kT_s)$ and $y_i^k = y_i(t_0 + kT_s)$, and point O_i is moving at a known constant velocity $v_i^k \triangleq [v_{i,x}^k\, v_{i,y}^k]^T$.

To develop a criterion for collision avoidance, we define the robot velocity relative to that of the ith object as

$$v_{r,i}^k \triangleq v_r - v_i^k = \begin{bmatrix} v_{r,i,x}^k \\ v_{r,i,y}^k \end{bmatrix} = \begin{bmatrix} \dot{x} - v_{i,x}^k \\ \dot{y} - v_{i,y}^k \end{bmatrix}. \tag{8}$$

Using the relative velocity, Fig. 5 is transformed into Fig. 6 in which the object is "static." According to Fig. 6, the collision-avoidance criterion in the y–x plane should be: for $x_i' \in [\underline{x}_i', \bar{x}_i']$ with $\underline{x}_i' = x_i^k - r_i - R$ and $\bar{x}_i' = x_i^k + r_i + R$, we have

$$\left(y_i' - y_i^k\right)^2 + \left(x_i' - x_i^k\right)^2 \geq (r_i + R)^2$$

where $\tau = t - (t_0 + kT_s)$ for $t \in [t_0 + kT_s, t_f]$, $x_i' = x - v_{i,x}^k\tau$, and $y_i' = y - v_{i,y}^k\tau$ (which are the predictive velocities for the future). Note that the circle of radius $r_i + R$ for collision avoidance can be placed around the center of either the robot or the ith object. If the circle is placed around the robot, the collision-avoidance criterion in the y–x plane becomes: whenever $x_i^k \in [x_i' - r_i - R, x_i' + r_i + R]$, as follows:

$$\left(y_i' - y_i^k\right)^2 + \left(x_i' - x_i^k\right)^2 \geq (r_i + R)^2 \tag{9}$$

where τ, x_i', and y_i' are defined as before.

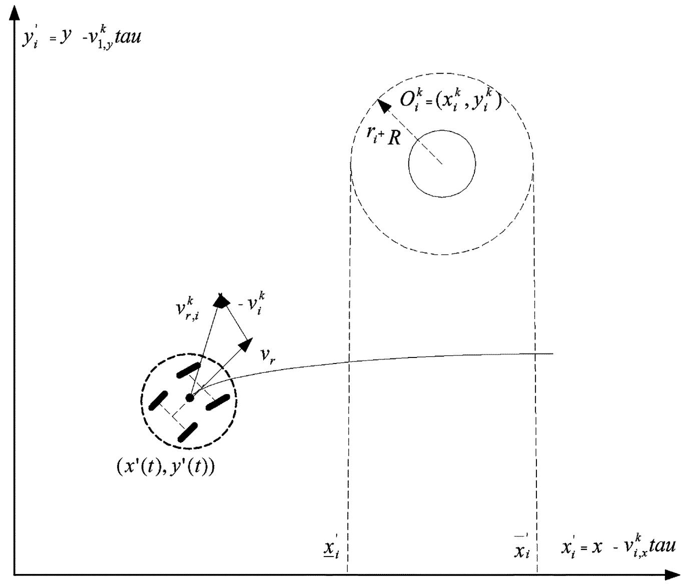

Fig. 6. Relative velocity of the robot with respect to the ith obstacle.

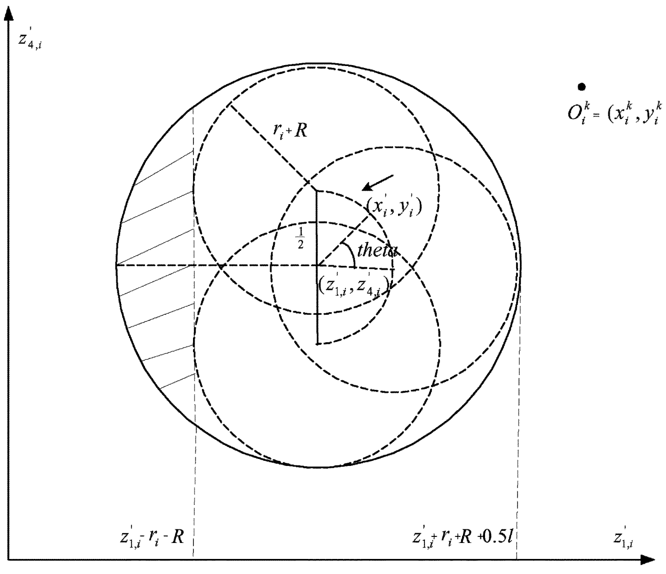

Fig. 7. Illustration of the collision-avoidance criterion in the transformed plane.

It follows from state transformation (2) that, given any steerable path $z_4 = F(z_1)$, the corresponding feasible path in the x-y plane is

$$y = F(x - 0.5l\cos(\theta)) + 0.5l\sin(\theta).$$

Thus, the corresponding collision-avoidance criterion in the transformed $z_4 - z_1$ space is: whenever $x_i^k \in [z'_{1,i} + 0.5l\cos(\theta) - r_i - R, z'_{1,i} + 0.5l\cos(\theta) + r_i + R]$, we have

$$\left(z'_{4,i} + \frac{l}{2}\sin(\theta) - y_i^k\right)^2 + \left(z'_{1,i} + \frac{l}{2}\cos(\theta) - x_i^k\right)^2 \geq (r_i + R)^2 \quad (10)$$

where $z'_{1,i} = z_1 - v^k_{i,x}\tau$ and $z'_{4,i} = z_4 - v^k_{i,y}\tau$.

Note that, although θ can be determined from z_3 and z_3 can be obtained as a result of applying *Lemma 1*, exact mapping from z to (x, y, θ) should not be used to numerically solve the problem of trajectory planning by imposing criterion (10). Instead, we choose to develop a new criterion only in terms of z_1 and z_4 (or z'_1 and z'_4) so that an analytical solution can be found for the problem of trajectory planning. To this end, note that all possible locations of point (x'_i, y'_i) are on the right semicircle centered at $(z'_{1,i}, z'_{4,i})$ and of radius $l/2$ for $\theta \in [-\pi/2, \pi/2]$. As shown in Fig. 7, plotting a family of circles of radius $(r_i + R)$ along the right semicircle renders the region from which the center of the ith object must stay clear, and the region is completely covered by the *unshaded* portion of the circle centered at $(z'_{1,i}, z'_{4,i})$ and of radius $(r_i+R+l/2)$. Mathematically, the proposed collision-avoidance criterion in the $z_4 - z_1$ plane is

$$\left(z'_{4,i} - y_i^k\right)^2 + \left(z'_{1,i} - x_i^k\right)^2 \geq \left(r_i + R + \frac{l}{2}\right)^2 \quad (11)$$

provided that

$$x_i^k \in [z'_{1,i} - r_i - R, z'_{1,i} + 0.5l + r_i + R]. \quad (12)$$

It is apparent from Fig. 7 that criterion (11) implies criteria (10) or (9). Once a steering method is chosen, the time interval during which criterion (11) should be imposed to avoid collision can be found from (12). That is, the proposed collision-avoidance scheme has two parts: time criterion (12) and geometrical criterion (11).

C. A Feasible Collision-Free Trajectory Parameterization and Solution

A specific candidate class of feasible, collision-free trajectories are parameterized as

$$z_4(z_1) = F(z_1) = a^k f(z_1) \quad (13)$$

where $a^k = [a_0^k, a_1^k, \ldots, a_6^k]$ is a constant vector to be determined, and $f(z_1) = [1, z_1(t), (z_1(t))^2, \ldots, (z_1(t))^6]^T$ is the vector composed of basis functions of $z_1(t)$. As discussed in *Remark 3.2*, the minimum order of polynomial-type feasible trajectories is fifth, and hence the class of sixth-order polynomials in (13) is of lowest order for collision avoidance under the boundary conditions if such a solution exists.

Our proposed method is to utilize the class of polynomials in (13) to parameterize motion trajectories. After imposing all of the boundary conditions in (5) and (6), the class of feasible trajectories have their polynomial parameterizations in terms of only one parameter a_6^k to be determined. This last parameter is then determined by using the proposed collision-avoidance criteria in (12) and (11). Because obstacles may be dynamically moving, the solution to a feasible and collision-free trajectory and its corresponding steering controls are solved once within time interval $t \in (t_0 + kT_s, t_0 + (k+1)T_s]$ and updated with respect to k, where T_s is the sampling period.

The following theorem is the main result of the paper, and it provides an analytical solution to the problem of finding a feasible collision-free trajectory. It is intuitively clear from the discussions in Section III-B that the third item in *Assumption 1* ensures solvability. Note that, in *Assumption 1*, the kth changing speeds of objects denoted by v_i^k need not to be known until $t = t_0 + kT_s$.

Assumption 1: Consider a nonholonomic car-like robot of model (1) and operating in the presence of circular moving obstacles that are centered at O_i and of radius r_i, for $i = 1, \ldots, n_o$. Assume the following.

- Boundary conditions, $q^0 = [x_0, y_0, \theta_0, \phi_0]^T$ and $q^f = [x_f, y_f, \theta_f, \phi_f]^T$ with $\phi_0 = \phi_f = 0$, are defined by (5) and (6), and they satisfy the conditions $x_0 - (l/2)\sin(\theta_0) \neq x_f - (l/2)\sin(\theta_f)$ and $|\theta_0 - \theta_f| < \pi$.
- Let $t_f = t_0 + T$ and T be the time for the mobile robot to complete its maneuver and T_s be the sampling period such that $\bar{k} = T/T_s$ is an integer, that centers of objects O_i are located at (x_i^k, y_i^k) at $t = t_0 + kT_s$, and that these objects are all moving with known constant velocities $v_i^k \overset{\triangle}{=} [v_{i,x}^k \, v_{i,y}^k]^T$ for $t \in [t_0 + kT_s, t_0 + (k+1)T_s)]$.
- For any given $k \in \{0, \ldots, \bar{k} - 1\}$, the free space is connected in the presence of unshaded circular regions given by that in Fig. 7 but located at O_i^k and of radius $r_i + R + 0.5l$, and the connectivity is with respect to "initial condition" (z_1^k, z_4^k) and "terminal condition" (z_1^{k+1}, z_4^{k+1}), where $z_i^k = z_i(t_0 + kT_s)$. Also, in relation to the free space and robot's sensing range, the robot's speed can be made faster than those of the objects.

Theorem 1: Under *Assumption 1*, a collision-free path can be generated analytically by undertaking the following steps.

Step 1) Select coordinates (x, y) of the working space such that $\theta \neq \pi/2$, apply state and input transformations (2) and (3), determine the corresponding boundary conditions $z^0 = [z_1^0, z_2^0, z_3^0, z_4^0]^T$ and $z^f = [z_1^f, z_2^f, z_3^f, z_4^f]^T$, and obtain the dynamics in chained form (4).

Step 2) For $k = 0, \ldots, \bar{k} - 1$, determine recursively constants a_6^k by ensuring the following second-order inequality (or inequalities): $\forall i \in \{1, \ldots, n_o^k\}$ where n_o^k is the number of objects within the sensing range[2] during time interval $t \in [t_0 + kT_s, t_0 + (k+1)T_s)]$ and $n_o^k \leq n_o$ as follows:

$$\min_{t \in [\underline{t}_i^*, \bar{t}_i^*]} g_2(z_1(t), k) \left(a_6^k\right)^2 + g_{1,i}(z_1(t), k, \tau) a_6^k + g_{0,i}(z_1(t), k, \tau)|_{\tau = t - t_0 - kT_s} \geq 0 \quad (14)$$

where $[\underline{t}_i^*, \bar{t}_i^*] \subset [t_0 + kT_s, t_f]$ is the time interval (if it exists[3]) during which

$$x_i^k \in \left[z_1(t) - v_{i,x}^k \tau - r_i - R, z_1(t) - v_{i,x}^k \tau + 0.5l + r_i + R\right]. \quad (15)$$

In (14), functions $z_1(t), g_2(\cdot), g_{1,i}(\cdot)$, and $g_{0,i}(\cdot)$ are defined as follows: for $k > 0$

$$\begin{aligned}
x_i^0 &= x_i(t_0)\\
y_i^0 &= y_i(t_0)\\
x_i^k &= x_i^0 + T_s \sum_{j=0}^{k-1} v_{i,x}^j\\
y_i^k &= y_i^0 + T_s \sum_{j=0}^{k-1} v_{i,y}^j\\
z_4^0 &= y_0 - \frac{l}{2}\sin(\theta_0)\\
z_3^0 &= \tan(\theta_0)\\
z_2^0 &= 0
\end{aligned}$$

$$z_1(t) = z_1^k + \frac{z_1^f - z_1^0}{T}(t - t_0 - kT_s) \quad \forall t \in [t_0 + kT_s, t_f] \quad (16)$$

$$z_1^k = z_1^0 + \frac{k\left(z_1^f - z_1^0\right)}{\bar{k}} \quad (17)$$

$$z_3^k = z_3^{k-1} + \frac{z_1^f - z_1^0}{\bar{k}} z_2^{k-1} + \frac{z_1^f - z_1^0}{T} \times \int_{t_0+(k-1)T_s}^{t_0+kT_s} \int_{t_0+(k-1)T_s}^{s} v_{c2}^{k-1}(\lambda)\, d\lambda\, ds \quad (18)$$

$$z_4^k = z_4^{k-1} + \frac{z_1^f - z_1^0}{\bar{k}} z_3^{k-1} + \frac{T_s^2}{2}\left(\frac{z_1^f - z_1^0}{T}\right)^2 \times z_2^{k-1} + \left(\frac{z_1^f - z_1^0}{T}\right)^2 \int_{t_0+(k-1)T_s}^{t_0+kT_s} \int_{t_0+(k-1)T_s}^{\tau} \times \int_{t_0+(k-1)T_s}^{s} v_{c2}^{k-1}(\lambda)\, d\lambda\, ds\, d\tau \quad (19)$$

$$z_2^k = z_2^{k-1} + \int_{t_0+(k-1)T_s}^{t_0+kT_s} v_{c2}^{k-1}(\lambda)\, d\lambda \quad (20)$$

$$Y^k = \begin{bmatrix} z_4^k \\ z_3^k \\ z_2^k \\ y_f - \frac{l}{2}\sin(\theta_f) \\ \tan(\theta_f) \\ 0 \end{bmatrix}, \quad A^k = \begin{bmatrix} \left(z_1^k\right)^6 \\ 6\left(z_1^k\right)^5 \\ 30\left(z_1^k\right)^4 \\ \left(z_1^f\right)^6 \\ 6\left(z_1^f\right)^5 \\ 30\left(z_1^f\right)^4 \end{bmatrix} \quad (21)$$

$$B^k = \begin{bmatrix} 1 & z_1^k & \left(z_1^k\right)^2 & \left(z_1^k\right)^3 & \left(z_1^k\right)^4 & \left(z_1^k\right)^5 \\ 0 & 1 & 2z_1^k & 3\left(z_1^k\right)^2 & 4\left(z_1^k\right)^3 & 5\left(z_1^k\right)^4 \\ 0 & 0 & 2 & 6z_1^k & 12\left(z_1^k\right)^2 & 20\left(z_1^k\right)^3 \\ 1 & z_1^f & \left(z_1^f\right)^2 & \left(z_1^f\right)^3 & \left(z_1^f\right)^4 & \left(z_1^f\right)^5 \\ 0 & 1 & 2z_1^f & 3\left(z_1^f\right)^2 & 4\left(z_1^f\right)^3 & 5\left(z_1^f\right)^4 \\ 0 & 0 & 2 & 6z_1^f & 12\left(z_1^f\right)^2 & 20\left(z_1^f\right)^3 \end{bmatrix} \quad (22)$$

$$\begin{aligned}
&\underline{f}(z_1(t))\\
&\quad = [1\, z_1(t)\, (z_1(t))^2\, (z_1(t))^3\, (z_1(t))^4\, (z_1(t))^5]\\
&g_2(z_1(t), k)\\
&\quad = [(z_1(t))^6 - \underline{f}(z_1(t))(B^k)^{-1}A^k]^2\\
&g_{1,i}(z_1(t), k, \tau)\\
&\quad = 2[(z_1(t))^6 - \underline{f}(z_1(t))(B^k)^{-1}A^k]\\
&\qquad \times \left[\underline{f}(z_1(t))(B^k)^{-1}Y^k - y_i^k - v_{i,y}^k \tau\right]\\
&g_{0,i}(z_1(t), k, \tau)
\end{aligned}$$

[2]Within the time interval $t \in [t_0 + kT_s, t_0 + (k+1)T_s)]$, n_o^k will change if an object moves into or out of the sensing range. When n_o^k changes, the path and its corresponding steering controls need to be updated. As the default, one could choose to update only if n_o^k increases.

[3]If the interval does not exist for some or all i, inequality (14) is not needed for those objects.

$$= \left[\underline{f}(z_1(t))(B^k)^{-1}Y^k - y_i^k - v_{i,y}^k\tau\right]^2 + \left(z_1(t) - x_i^k - v_{i,x}^k\tau\right)^2 - (r_i + R + 0.5l)^2.$$

Step 3) A feasible, collision-free path of form (13) in the transformed state is found by solving a^k according to

$$a^k = \left[a_0^k\, a_1^k\, a_2^k\, a_3^k\, a_4^k\, a_5^k\, a_6^k\right]$$
$$\left[a_0^k, a_1^k, a_2^k, a_3^k, a_4^k, a_5^k\right]^T = (B^k)^{-1}\left(Y^k - A^k a_6^k\right). \quad (23)$$

Step 4) The steering inputs to achieve path (13) are given by, for $t \in (t_0 + kT_s, t_0 + (k+1)T_s]$, $v_{c1}(t) = v_{c1}^k$, and $v_{c2}(t) = v_{c2}^k(t)$, where

$$v_{c1}^k(t) = \frac{z_1^f - z_1^0}{T} \quad (24)$$
$$\begin{aligned} v_{c2}^k(t) &= 6\left[a_3^k + 4a_4^k z_1^k + 10a_5^k\left(z_1^k\right)^2 + 20a_6^k\left(z_1^k\right)^3\right]v_{c1}^0 \\ &+ 24\left[a_4^k + 5a_5^k z_1^k + 15a_6^k\left(z_1^k\right)^2\right](t - t_0 - kT_s)\left(v_{c1}^0\right)^2 \\ &+ 60\left(a_5^k + 6a_6^k z_1^k\right)(t - t_0 - kT_s)^2\left(v_{c1}^0\right)^3 \\ &+ 120a_6^k(t - t_0 - kT_s)^3\left(v_{c1}^0\right)^4. \end{aligned} \quad (25)$$

Step 5) The corresponding feasible, collision-free Cartesian trajectory is given by $y = F(x - 0.5l\cos(\theta)) + 0.5l\sin(\theta)$, where θ can be found in closed form from state transformation (2) under steering inputs (24) and (25) together with control mapping (3).

Proof: The proof, provided below according to the statements in the theorem, is done recursively for time intervals $t \in (t_0 + kT_s, t_0 + (k+1)T_s]$, with boundary conditions in (5) and (6), and with intermediate boundary conditions in equations from (17) to (20).

Step 1) Obvious from the discussions in Section II.

Steps 2) and 3) Consider the class of candidate trajectories in (13). Starting with initial conditions in (5) and given intermediate boundary conditions at time instant $t_0 + (k-1)T_s$ for $k \geq 1$, the intermediate boundary conditions at time instant $t_0 + kT_s$ can be obtained by directly integrating (4), and the results are given in equations from (17) to (20). Then, applying boundary conditions from (17) to (20) and (6) to (13), we have

$$B^k\left[a_0^k, a_1^k, a_2^k, a_3^k, a_4^k, a_5^k\right]^T = Y^k - A^k a_6^k$$

which renders (23). Note that matrix B^k in (22) is nonsingular as long as $z_1^0 \neq z_1^f$. Hence, trajectories satisfying all of the boundary conditions are parameterized in terms of a_6^k as

$$z_4 = \begin{bmatrix} (B^k)^{-1}\left(Y^k - A^k a_6^k\right) \\ a_6^k \end{bmatrix}^T f(z_1).$$

Substituting the above equation into (11) yields (14), a second-order polynomial inequality in a_6^k.

Step 4) To determine the steering control inputs, let

$$\begin{aligned} v_{c1}^k(t) &= C \\ v_{c2}^k(t) &= C_0^k + C_1^k(t - t_0 - kT_s) + C_2^k(t - t_0 - kT_s)^2 \\ &+ C_3^k(t - t_0 - kT_s)^3 \end{aligned}$$

where C and $C_j^k, j = 0, 1, 2, 3$ are constants. Directly integrating (4) yields

$$\begin{aligned} z_1(t) &= z_1^k + C(t - t_0 - kT_s) \\ z_2(t) &= z_2^k + C_0^k(t - t_0 - kT_s) + \frac{C_1^k}{2}(t - t_0 - kT_s)^2 \\ &+ \frac{C_2^k}{3}(t - t_0 - kT_s)^3 + \frac{C_3^k}{4}(t - t_0 - kT_s)^4 \\ z_3(t) &= z_3^k + Cz_2^k(t - t_0 - kT_s) + \frac{CC_0^k}{2}(t - t_0 - kT_s)^2 \\ &+ \frac{CC_1^k}{6}(t - t_0 - kT_s)^3 + \frac{CC_2^k}{12}(t - t_0 - kT_s)^4 \\ &+ \frac{CC_3^k}{20}(t - t_0 - kT_s)^5 \\ z_4(t) &= z_4^k + Cz_3^k(t - t_0 - kT_s) + \frac{C^2 z_2^k}{2}(t - t_0 - kT_s)^2 \\ &+ \frac{C^2 C_0^k}{6}(t - t_0 - kT_s)^3 + \frac{C^2 C_1^k}{24}(t - t_0 - kT_s)^4 \\ &+ \frac{C^2 C_2^k}{60}(t - t_0 - kT_s)^5 + \frac{C^2 C_3^k}{120}(t - t_0 - kT_s)^6 \end{aligned} \quad (26)$$

for $t \in (t_0 + kT_s, t_0 + (k+1)T_s]$. It is obvious that we have

$$C = \frac{z_1^f - z_1^0}{T}.$$

On the other hand, substituting $z_1(t) = z_1^k + C(t - t_0 - kT_s)$ into $z_4 = a^k f(z_1)$ yields

$$\begin{aligned} z_4(t) &= b_0 + b_1(t - t_0 - kT_s) + b_2(t - t_0 - kT_s)^2 \\ &+ b_3(t - t_0 - kT_s)^3 + b_4(t - t_0 - kT_s)^4 \\ &+ b_5(t - t_0 - kT_s)^5 + b_6(t - t_0 - kT_s)^6 \end{aligned} \quad (27)$$

where

$$\begin{aligned} b_0 &= \sum_{i=0}^{6} a_i^k (z_1^k)^i \\ b_1 &= a_1^k C + 2a_2^k C z_1^k + 3a_3^k C(z_1^k)^2 + 4a_4^k C(z_1^k)^3 \\ &+ 5a_5^k C(z_1^k)^4 + 6a_6^k C(z_1^k)^5 \\ b_2 &= a_2^k C^2 + 3a_3^k C^2 z_1^k + 6a_4^k C^2(z_1^k)^2 + 10a_5^k C^2(z_1^k)^3 \\ &+ 15a_6^k C^2(z_1^k)^4 \\ b_3 &= a_3^k C^3 + 4a_4^k C^3 z_1^k + 10a_5^k C^3(z_1^k)^2 + 20a_6^k C^3(z_1^k)^3 \\ b_4 &= a_4^k C^4 + 5a_5^k C^4 z_1^k + 15a_6^k C^4(z_1^k)^2 \\ b_5 &= a_5^k C^5 + 6a_6^k C^5 z_1^k \\ b_6 &= a_6^k C^6. \end{aligned}$$

Then, in light of (23), we can solve for constants C_i^k in $v_{c2}^k(t)$ by comparing (27) and (26). The result renders the steering inputs in (24) and (25).

Step 5) It is obvious from the discussions in the previous sections. ♢

Remark 3.4: The theorem can be used for both trajectory planning and real-time trajectory replanning. If boundary steering angles (ϕ_0 and ϕ_f) and the corresponding boundary curvatures are nonzero (in a more general case), *Theorem 1* still holds except that Y^k in (21) should be set to be

$$Y^k = \left[z_4^k, z_3^k, z_2^k, y_f - \frac{l}{2}\sin(\theta_f), \tan(\theta_f), \frac{\tan(\phi_f)}{l\cos^3(\theta_f)}\right]^T$$

where $z_2^0 = \tan(\phi_0)/(l\cos^3(\theta_0))$. ♢

Remark 3.5: Due to the boundary conditions imposed in (17)–(20) and those in matrices Y_k, A_k, and B_k, the collision-free path found in the transformed plane of z_1 and z_4 is always smooth (more specifically, differentiable at least twice). On the other hand, steering inputs in (24) and (25) are piecewise continuous. Consequently, the resulting collision-free trajectory in the original x–y plane is also differentiable. ♢

Remark 3.6: It is clear from the proof of the theorem that a quintic polynomial function in the transformed space can satisfy the boundary conditions specified in (5) and (6). Therefore, for collision avoidance, polynomial (13) is of the minimum order necessary. Polynomials higher than sixth order provide additional freedom in the design and may potentially yield better performance, which is a topic of future study. ♢

Remark 3.7: If only constant velocity objects are present, one can simply choose $T_s = t_f$. In this case, we have $\bar{k} = 1$ and $k = 0$; intermediate boundary conditions are no longer needed; and steering inputs in (24) and (25) are continuous. In particular, collision-free criterion (14) and the z-plane solution to trajectory planning become

$$g_2(z_1, k)(a_6)^2 + g_{1,i}(z_1, k, \tau)a_6 + g_{0,i}(z_1, k, \tau) \geq 0, \quad i = 1, \ldots, n_o \tag{28}$$

and

$$z_4 = \underline{f}(z_1(t))(B^0)^{-1}(Y^0 - A^0 a_6) + a_6(z_1(t))^6 \tag{29}$$

where

$$g_2(z_1, k) = \left[z_1^6 - \underline{f}(z_1(t))(B^0)^{-1}A^0\right]^2. \tag{30}$$

Nonetheless, even if all objects keep moving at constant speeds, choosing $\bar{k} > 1$ may improve performance as it can approximate polynomials higher than sixth order. ♢

Remark 3.8: In the presence of dynamically moving objects whose speeds change rapidly, T_s should be chosen to be sufficiently small. On the other hand, if the robot is operating at the same speed, T_s becoming smaller implies that $|z_1^k - z_1^f|$ becomes smaller as k approaches $\bar{k} - 1$. This in turn makes matrix B^k in (22) closer to being singular. Thus, it is necessary for computational efficiency and robustness that T_s is not too small. ♢

Remark 3.9: It is apparent from steering inputs in (24) and (25) that, the larger the value of t_f is chosen, the smaller the values of steering inputs become. As such, t_f should be chosen to meet constraints on steering inputs and/or their rates of change. On the other hand, the collision-avoidance capability will be physically curtailed if the values of steering inputs are limited, as will be shown in the next remark. To avoid dynamically moving objects, a robot must maneuver fast enough with respect to motions of the objects. In other words, as illustrated by the proposed criterion, the robot has to be able to properly adjust its relative velocities with respect to the moving objects. ♢

Remark 3.10: Assumption 1 needs to be and can be relaxed in two ways. First, since the algorithm is applied iteratively, the condition on connectivity of the free space between "initial condition" (z_1^k, z_4^k) and "terminal condition" (z_1^{k+1}, z_4^{k+1}) can be reduced to that between the initial condition (z_1^0, z_4^0) and the terminal condition (z_1^f, z_4^f). Furthermore, if connectivity of the free space is lost for some finite values of k, there is no solution for a_6^k during those periods of time but, so long as connectivity is eventually recovered and maintained afterwards, a desired trajectory can be found. In other words, the proposed algorithm solves the trajectory-planning problem if there is a solution and it is flexible enough to handle issues that may be encountered in practice. ♢

Remark 3.11: Computationally, the proposed approach requires that inequality (14) be solved for at most $n_o \times \bar{k}$ times, where n_o is the number of the objects and $\bar{k}$ is the number of changes detected in the velocities of the objects. The equality version of (14) has two closed-form solutions, and its computational complexity depends only on the number of boundary conditions imposed (that is, matrix multiplications are six-dimensional). Therefore, the proposed algorithm is well suited for real-time implementation.

In comparison, a tree search routine would depend on a product of n_o, n_t, n_{z_1}, and n_{z_4}, where n_t, n_{z_1}, and n_{z_4} are the numbers of grids along the t, z_1, and z_4 axes, respectively. Since it is usual that $n_t \gg \bar{k}$ and $n_{z_1}, n_{z_4} \gg 1$, a tree search algorithm is rapidly growing and, if implemented online, is, in comparison, much more computationally intensive. ♢

Remark 3.12: It is noted that, in the proposed steering control (24), $v_{c1}(t)$ is a constant. Under this choice, we know from (3) that Cartesian x-directional speed is constant while y-directional speed is not. To have time-varying speeds in both directions, one can simply make $v_{c1}(t)$ piecewise constant as

$$v_{c1}(t) = C^{k-1}, \quad t \in [t_0 + (k-1)T_s, t_0 + kT_s), \quad k = 1, \ldots, \bar{k} \tag{31}$$

where C^{k-1} is constant and satisfies the condition

$$\int_{t_0}^{t_f} v_{c1}(\lambda)\, d\lambda = z_1^f - z_1^0.$$

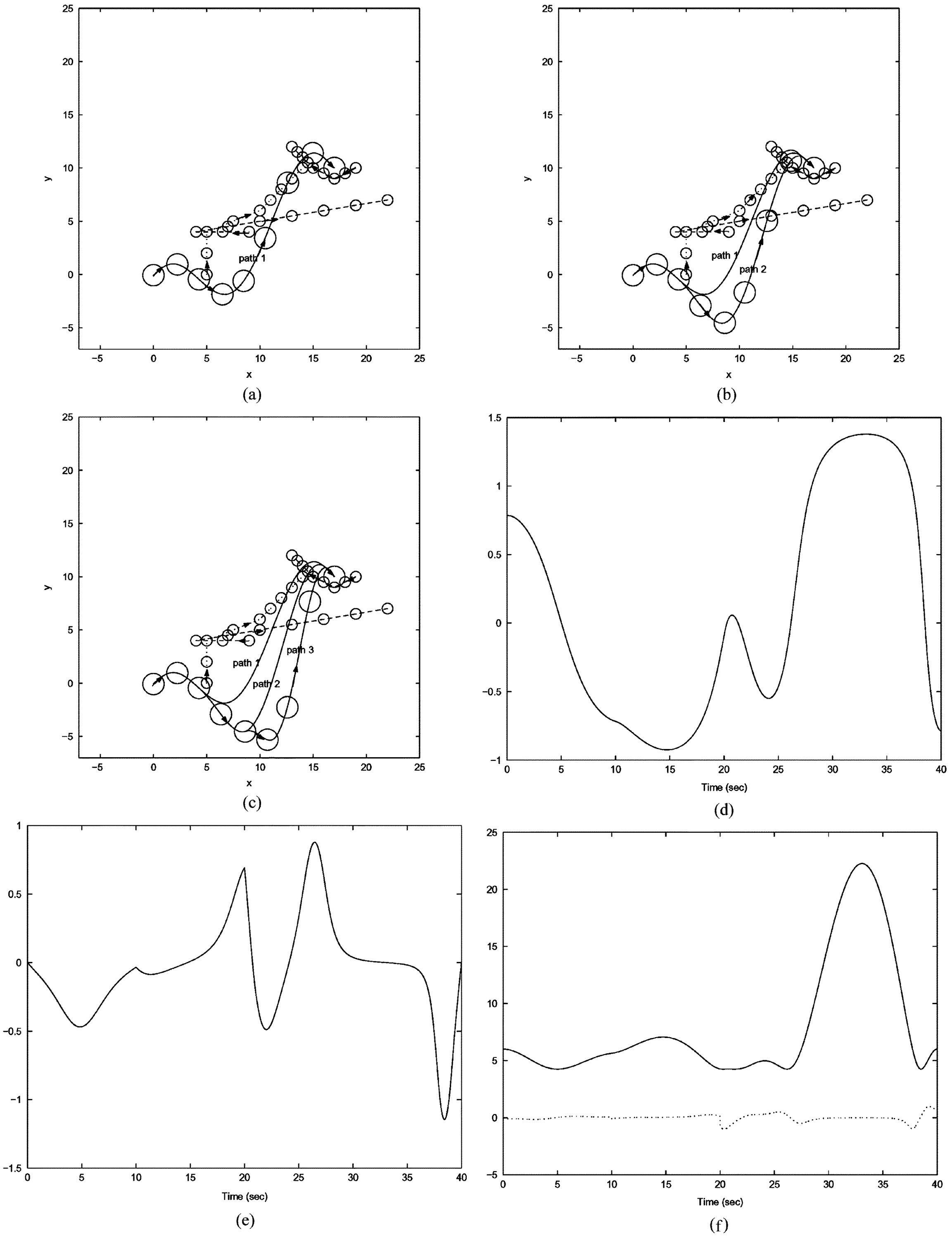

Fig. 8. Simulation results under $|a_6^k|$ of smaller magnitude. (a) Paths of robot and obstacles. (b) Paths of robot and obstacles. (c) Paths of robot and obstacles. (d) Entire trajectory of $\theta(t)$. (e) Entire trajectory of $\phi(t)$. (f) Steering controls: $u_1(t)$—solid line and u_2—dashed line.

Upon replacing $v_{c1}(t)$ in (24) by (31), it is straightforward to show that $v_{c2}(t)$ has the same expression as that in (25) and that *Theorem 1* holds, except that (16)–(19) are replaced by the following ones, respectively:

$$
\begin{aligned}
z_1(t) &= z_1^k + \int_{t_0+kT_s}^{t} v_{c1}(s)\, ds \quad \forall t \in [t_0 + kT_s, t_f] \\
z_1^k &= z_1^{k-1} + T_s C^{k-1}
\end{aligned}
$$

$$
\begin{aligned}
z_3^k =& z_3^{k-1} + T_s C^{k-1} z_2^{k-1} \\
&+ C^{k-1} \int_{t_0+(k-1)T_s}^{t_0+kT_s} \int_{t_0+(k-1)T_s}^{s} v_{c2}^{k-1}(\lambda) d\lambda ds \\
z_4^k =& z_4^{k-1} + T_s C^{k-1} z_3^{k-1} + \frac{T_s^2}{2} C^{k-1} z_2^{k-1} + C^{k-1} \\
&\times \int_{t_0+(k-1)T_s}^{t_0+kT_s} \int_{t_0+(k-1)T_s}^{\tau} \int_{t_0+(k-1)T_s}^{s} v_{c2}^{k-1}(\lambda) d\lambda ds d\tau.
\end{aligned}
$$

An extension to obtain a smooth control input $v_{c1}(t)$ may also be done with little difficulty. $\diamondsuit$

Given the class of feasible trajectories in (13), collision avoidance in the presence of multiple moving objects depends upon solvability of inequality (14). To see that *Assumption 1* ensures solvability, note that $g_2(z_1,k) \geq 0$. If $g_2(z_1,k) \neq 0$, inequalities in (14) belong to a family of parabolic functions with openings upward in the plane of a_6^k versus t. Thus, as long as $g_2(z_1,k) \neq 0$, a solution to a_6^k always exists for any one object and, for the ith object, the solution is of the form $a_6^k \notin (\underline{c}_i, \bar{c}_i)$, where $\underline{c}_i$ and $\bar{c}_i$ are the two roots of the equality version of (14). In the presence of multiple objects, the final solution is given by $a_6^k \notin \bigcup_{i=1}^{n_o}(\underline{c}_i, \bar{c}_i)$, and it always yields at least one finite value for a_6^k unless the objects and their associated collision regions make the free space disconnected between "initial condition" (z_1^k, z_4^k) and "terminal condition" (z_1^{k+1}, z_4^{k+1}). On the other hand, if $g_2(z_1,k) = 0$, it follows that $g_{1,i}(z_1,k,\tau) = 0$ as well and that the choice of a_6^k does little to make inequality (14) valid.

To understand the condition of $g_2(z_1,k) = 0$ and its implications, consider the simpler case that $\bar{k} = 1$. It follows from (29) that

$$z_4 = \underline{f}(z_1(t))(B^0)^{-1}Y^0 + a_6\left[z_1^6 - \underline{f}(z_1(t))(B^0)^{-1}A^0\right].$$

Comparing with the expression of $g_2(z_1,k)$ in (30), we know that $g_2(z_1,k) = 0$ if, no matter what a_6 is chosen, the feasible trajectory is degenerated into a quintic polynomial. Clearly, this is impossible for the function $z_4 = af(z_1)$ except for the boundary points at which boundary conditions must hold for both fifth- and sixth-order polynomials, and hence, order degeneration occurs. At the boundary points, inequality (28) is not needed unless t_0 and/or t_f is in the interval $[\underline{t}_i^*, \bar{t}_i^*]$ as defined by (15). In the first case, collision has occurred already, and little can be done. In the second case, collision can be avoided by adjusting t_f (unless one of the objects stays close enough to z^f). Other than these two cases, $g_{0,i}(z_1,k,\tau) \geq 0$ and hence inequality (14) is valid even though $g_2(z_1,k) = 0$. For the general case of $\bar{k} > 1$, one can inductively conclude the third item in *Assumption 1* is the solvability condition as the planning algorithm is applied iteratively.

In the vicinity of but not exactly at the boundary conditions, $g_2(z_1,k)$ is close to being zero. If the interval (15) contains time instants corresponding to any of these points, one can form a proper solution to a_6 not by solving inequality (28), but by adjusting the robot speed and in turn the interval in (15) so that it no longer contains any value very close to the boundary time instants. In summary, for the trajectory-planning problem in the presence of multiple dynamically moving objects, a solution exists under *Assumption 1* and can be obtained using the theorem.

IV. Simulation

In this section, the proposed steering algorithm is simulated to illustrate its effectiveness. In the simulations, the following settings are used:

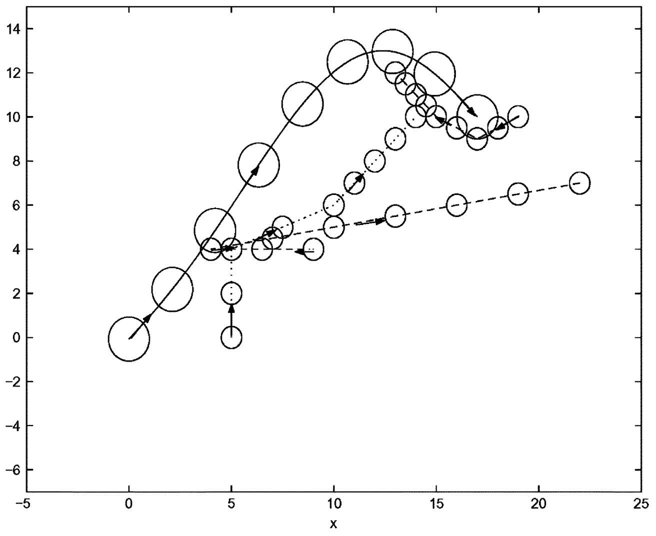

Fig. 9. Free-space path ($a_6 = 0$) versus trajectories of obstacle 1 (dotted line), obstacle 2 (dash–dotted line), and obstacle 3 (dashed line).

- Robot parameters: $R = 1$, $l = 0.8$, and $\rho = 0.2$.
- Boundary conditions: $q^0 = (0, 0, (\pi/4), 0)$ and $q^f = (17, 10, -(\pi/4), 0)$, where $t_0 = 0$ and $t_f = 40$ s.
- Moving obstacles: $n_0 = 3, O_1(t_0) = [5,0]^T, O_2(t_0) = [9,4]^T, O_3(t_0) = [19,10]^T$, and $r_i = 0.5$ for $i = 1,2,3$.
- Speeds of obstacles:

$$\begin{aligned}
&v_1^0 = [0, 0.4]^T, \quad v_1^1 = [0.5, 0.2]^T, \quad v_1^2 = v_1^3 = [0.2, 0.2]^T\\
&v_2^0 = [-0.5, 0]^T, \quad v_2^1 = [0.6, 0.1]^T, \quad v_2^2 = v_2^3 = [0.6, 0.1]^T\\
&v_3^0 = [-0.2, -0.1]^T, \quad v_3^1 = [-0.2, 0.1]^T\\
&v_3^2 = v_3^3 = [-0.1, 0.1]^T.
\end{aligned}$$

The sampling period is adaptively chosen according to speed changes detected, and hence $T_s = 10$ s.

- Baseline choices of design parameters: $R_s = 25$, which implies that the robot has enough sensor range to detect all three obstacles at all times (i.e., $n_o^k = 3$ for $k = 0, 1, 2, 3$).
- At each sampling instant, current steering controls are determined based on the motion of obstacles detected within the sensor range of the robot. Therefore, the baseline solution to the parameterized trajectory is given by

$$\begin{aligned}
a_6^0 &= 2.9659 * 10^{-5}\\
a_6^1 &= 1.0577 * 10^{-4}\\
a_6^2 &= a_6^3 = 0.0013
\end{aligned} \tag{32}$$

which are the solutions to the equality version of (14) in *Theorem 1* and of the smaller magnitude.

All quantities conform to a given unit system, for instance, meter, meter per second, etc. The case of the robot having a limited sensing range will be presented at the end of this section.

The simulation results under the solution of (32) are shown in Fig. 8(a)–(f). In Fig. 8(a)–(c), the paths of the robot and obstacles 1–3 are represented by the solid line, the dotted line, the dash–dotted line, and the dashed line, respectively.

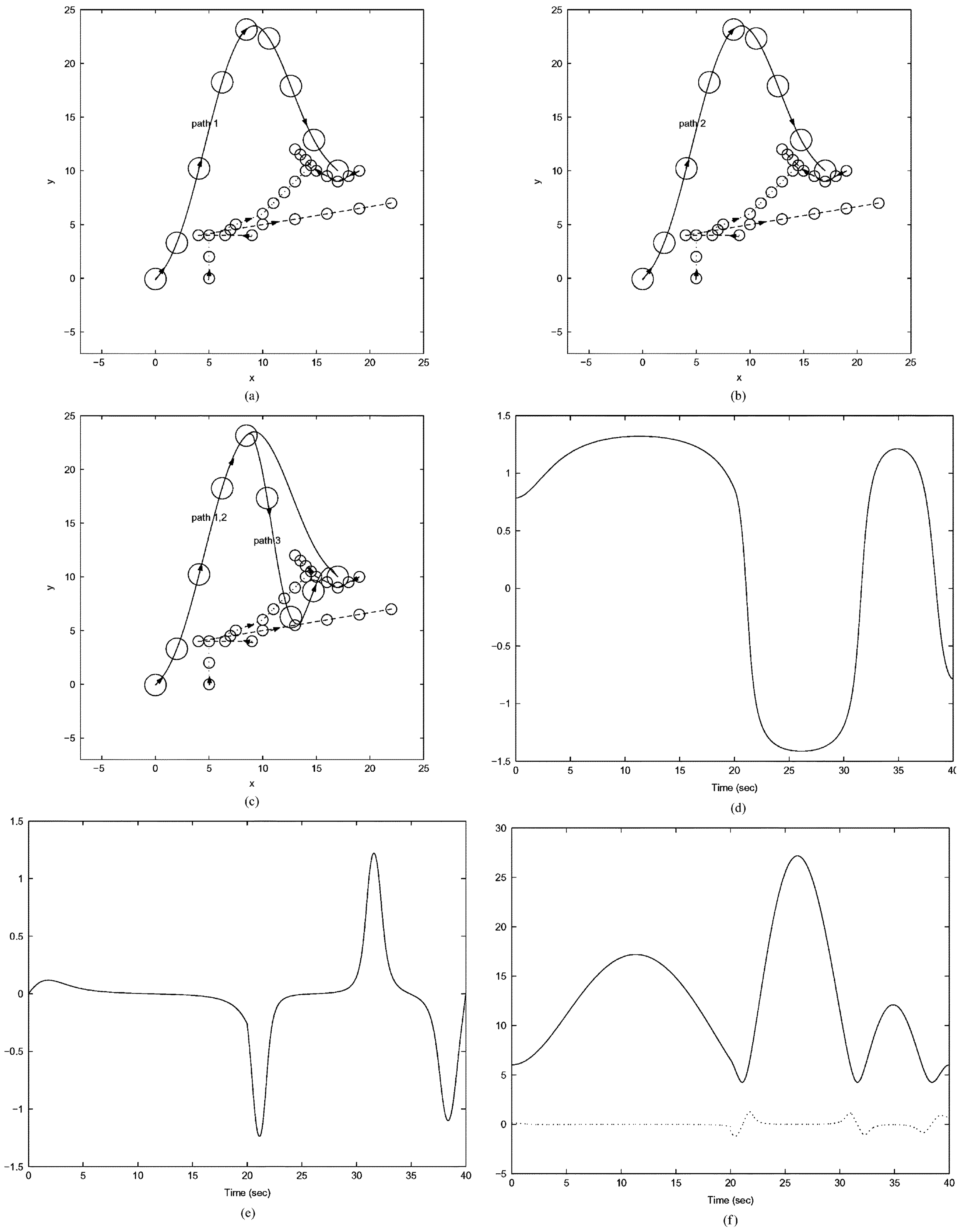

Fig. 10. Simulation results under $|a_6^k|$ of larger magnitude. (a) Paths of robot and obstacles. (b) Paths of robot and obstacles. (c) Paths of robot and obstacles. (d) Entire trajectory of $\theta(t)$. (e) Entire trajectory of $\phi(t)$. (f) Steering controls: $u_1(t)$—solid line and u_2—dashed line.

Fig. 8(a) provides the trajectory solution (called path 1) if the objects were to maintain their velocities at v_i^0 for $t = [0, T]$.

Fig. 8(b) shows the trajectory (called path 2 and compared to path 1) if the object velocities were to change from v_i^0 to v_i^1 but be kept at v_i^1 afterwards (for $t = [10, 40]$). Finally, Fig. 8(c) is the complete solution of the entire trajectory (called path 3 and compared to paths 1 and 2) by considering all of the speed changes of all objects.

In Fig. 8(a)–(c), positions of the robot and the three objects at $t = 0$ and every five seconds afterwards are marked by small circles along their corresponding trajectories, respectively. It is clear from Fig. 8(a) that, if path 1 is followed beyond $t = 10$, collisions between the robot and objects 1 and 3 will occur around $t = 30$ and $t = 35$ s, respectively. Similarly, Fig. 8(b) shows that if path 2 is followed beyond $t = 20$, a collision between the robot and object 3 will occur around $t = 35$ s. By design, path 3 is collision-free.

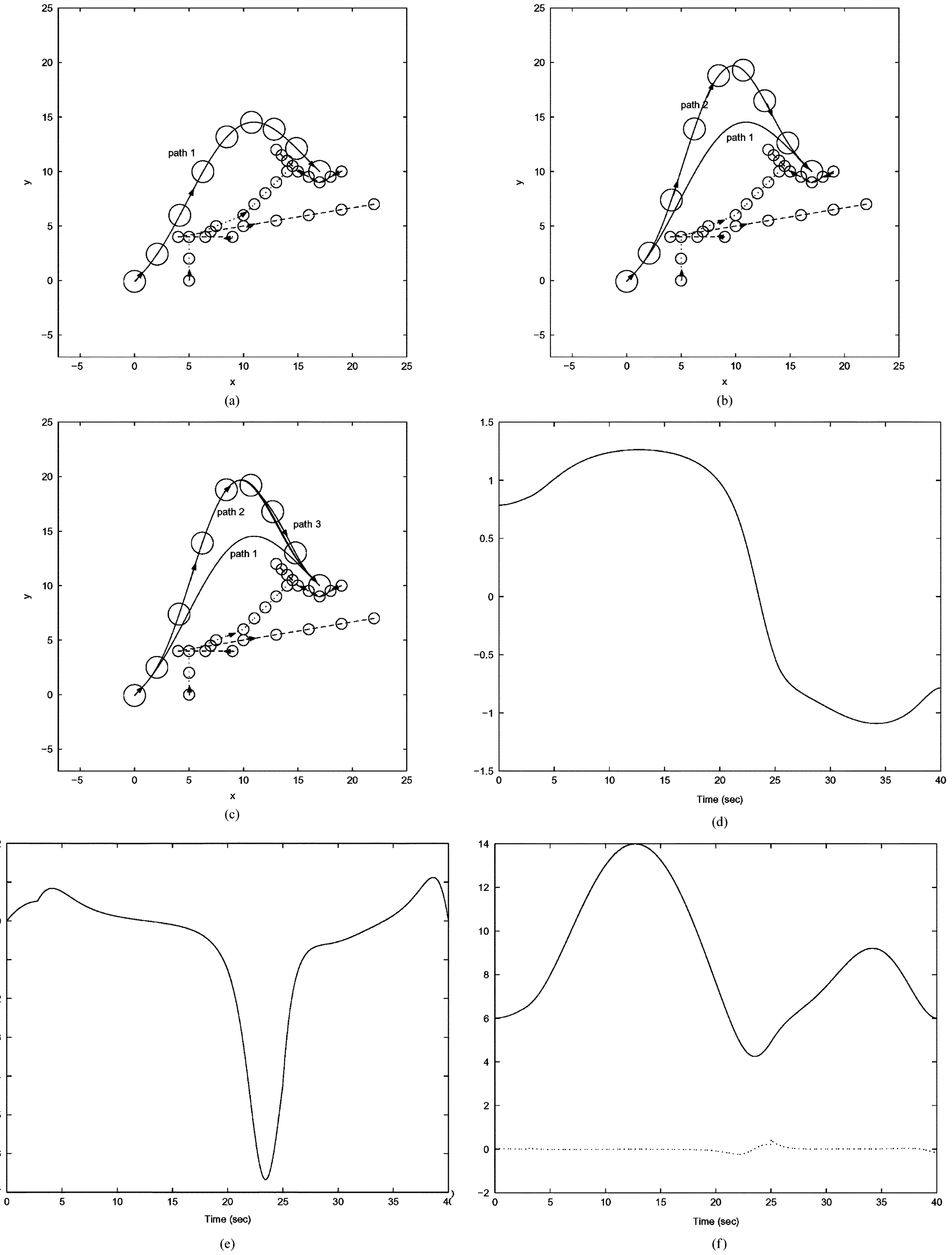

Fig. 11. Simulation results under $R_s = 7$. (a) Paths of robot and obstacles. (b) Paths of robot and obstacles. (c) Paths of robot and obstacles, (d) Entire trajectory of $\theta(t)$. (e) Entire trajectory of $\phi(t)$. (f) Steering controls: $u_1(t)$—solid line and u_2—dashed line.

Fig. 8(d)–(f) contains the entire trajectories of orientation angle, steering angle, and steering controls, respectively.

Piecewise solution (32) is found by solving the equality version of inequality (14). There are two solutions, and a_6^k in (32) is set to be that of smaller magnitude. This is because the smaller $|a_6^k|$ is, the closer the resulting trajectory is to the free-space trajectory (which is shown in Fig. 9 and has collisions with objects 1, 2, and 3 around $t = 10, 10, 35$ s, respectively). The solution of larger magnitude is given by

$$a_6^0 = -3.3343 * 10^{-5}$$
$$a_6^1 = -3.3343 * 10^{-5}$$
$$a_6^2 = a_6^3 = 0.0019.$$

For comparison purposes, the corresponding trajectories are plotted in Fig. 10(a)–(f). It turns out that, for the two-input four-state chained form, the swing direction of planned trajectories is determined by the sign of a_6^k: upwards if a_6^k is negative, and downwards if a_6^k is positive. On the other hand, if a_6^k is set to be that of larger magnitude, the resulting trajectory tends to have larger detour or swing, which may be undesirable. Therefore, a combination of a_6^k (in terms of their magnitudes and signs) may prove to be better in some situations.

For comparison purposes, let us finally consider the case that $R_s = 7$. In this case, the robot's sensor has a limited range so the robot detects the presence of objects 1, 2, and 3 intermittently. Sampling period T_s is chosen to account for speed changes of objects detected, and n_o^k is introduced to account for emergence and disappearance of various objects in the sensing range of the robot. It is obvious that, when T_s elapses or n_o^k increases, the proposed algorithm needs to be applied to update the trajectory and its corresponding steering controls. Thus, the only difference from the baseline simulation is that, in light of the limited sensing range, n_o^k is changing, and so are the planned trajectory and steering controls. Again, all of the following choices of a_6^k are those of smaller magnitude.

Given $R_s = 7$, the evolution of planned trajectories are plotted in Fig. 11(a)–(f). Specifically, at $t = 0, n_o^0 = 1$ (object 1 only) and applying the algorithm in *Theorem 1* yields $a_6^0 = -6.8863 * 10^{-6}$. The corresponding trajectory is shown by path 1 in Fig. 11(a), and it is kept until either T_s elapses or n_o^k changes. At time instant $t = 2.8$, object 2 is detected by the robot sensor, and n_o^0 becomes 2 (objects 1 and 2). Accordingly, using the same algorithm and at $t = 2.8, a_6^0$ is updated to be $a_6^0 = -3.0149 * 10^{-5}$, and for $t \in [2.8, 10]$ the robot will be commanded to follow path 2 given in Fig. 11(b). It is clear from Fig. 11(a) that, if the robot followed path 1 for the entire time interval $t \in [0, 10]$, a collision between the robot and object 2 will occur around $t = 8$ s.

The rest evolution of planned trajectories is conceptually the same. In particular, at $t = 10$, we have $n_o^1 = 2$ (objects 1 and 2) and hence $a_6^1 = -3.0149 * 10^{-5}$ is kept at the same value as that of preceding a_6^0. Within the interval $t \in [10, 20)$, objects 1 and 2 gradually move out of the sensing range, nonetheless the robot trajectory can remain path 2 in Fig. 11(b) (or, one could choose to replan the trajectory). At $t = 20, n_o^2 = 0$ (no object), the algorithm chooses to keep a_6^k as $a_6^2 = a_6^1$ so that the robot continues to follow path 2. Around $t = 25$, objects 1 and 3 are detected, n_o^2 is updated to be $n_o^2 = 2$, and $a_6^2 = -6.3247 * 10^{-4}$ is determined by the algorithm. Thus, for $t \in [25, 30]$ and $t \in [30, 40]$, the robot is commanded to follow path 3 plotted in Fig. 11(c).

V. Conclusion

In this paper, a new collision-avoidance paradigm is proposed to solve the problem of real-time trajectory generation. While the robotic platform is chosen to be a four-wheel car-like mobile vehicle, the proposed paradigm uses the chained form as the basic model and therefore is applicable to other nonholonomic systems. Based on a piecewise-constant polynomial parameterization of all feasible trajectories, the proposed scheme prevents any collision by checking a time criterion and then a geometrical criterion, and it yields analytical solutions to collision-free path(s) and the corresponding steering controls. The piecewise-constant representation of feasible trajectories and steering controls enables the proposed method to admit such changes in a dynamical environment as speed change of obstacles, limited sensor range (and the corresponding appearance and disappearance of obstacles), and resetting of terminal conditions. Solvability condition is explicitly found. Effectiveness of the proposed method are illustrated by simulation results.

References

[1] J. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer, 1998.

[2] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 501–518, Oct. 1992.

[3] J. Borenstein and Y. Koren, "The vector field histogram—Fast obstacle avoidance for mobile robots," *IEEE Trans. Robot. Automat.*, vol. 7, pp. 278–288, June 1991.

[4] J.-P. Laumond, *Robot Motion Planning and Control*. London, U.K.: Springer-Verlag, 1998.

[5] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC Press, 1994.

[6] H. J. Sussmann and W. Liu, "Limits of Highly Oscillatory Controls and the Approximation of General Paths by Admissible Trajectories," Rutgers Ctr. Systems and Control, Piscataway, NJ, Tech. Rep. SYSCON-91-02, Feb. 1991.

[7] M. Fliess, J. Levine, Ph. Martin, and P. Rouchon, "Flatness and defect of nonlinear systems: Introductory theory and examples," *Int. J. Contr.*, vol. 61, pp. 1327–1361, 1995.

[8] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Trans. Automat. Contr.*, vol. 38, pp. 700–716, May 1993.

[9] S. Monaco and D. Normand-Cyrot, "An introduction to motion planning under multirate digital control," in *Proc. 31st Conf. Decision and Control*, Tucson, AZ, Dec. 1992, pp. 1780–1785.

[10] D. Tilbury, R. M. Murray, and S. S. Sastry, "Trajectory generation for the n-trailer problem using goursat normal form," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 802–819, May 1995.

[11] C. Fernandes, L. Gurvits, and Z. Li, "Near-optimal nonholonomic motion planning for a system of coupled rigid bodies," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 450–463, Mar. 1994.

[12] J. A. Reeds and R. A. Shepp, "Optimal paths for a car that goes both forward and backward," *Pacific J. Math.*, vol. 145, pp. 367–393, 1990.

[13] H. J. Sussmann and G. Tang, "Shortest Paths for the Reeds–Shepp Car: A Worked Out Example of the Use of Geometric Techniques in Nonlinear Optimal Control," Rutgers Univ., Piscataway, NJ, Tech. Rep. SYSCON-91-10, 1991.

[14] S. Sundar and Z. Shiller, "Optimal obstacle avoidance based on the Hamilton–Jacobi–Bellman equation," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 305–310, Apr. 1997.

[15] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control*, 2nd ed. New York: Hemisphere, 1975.

[16] Z. Qu and J. R. Cloutier, "A new suboptimal control design for cascaded nonlinear systems," *Optimal Contr.: Applicat. Methods*, vol. 23, pp. 303–328, Nov. 2002.

[17] J.-P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, "A motion planner for nonholonomic mobile robots," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 577–593, Oct. 1994.

[18] J. Barraquand and J.-C. Latombe, "Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles," in *Proc. IEEE Int. Conf. Robotics and Automation*, Sacramento, CA, Apr. 1991, pp. 2328–2335.

[19] A. W. Divelbiss and J. T. Wen, "A path space approach to nonholonomic motion planning in the presence of obstacles," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 443–451, June 1997.

[20] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," in *J. Assoc. Comput. Machinery*, vol. 40, 1993, pp. 1048–1066.

[21] S. Lavalle and J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, pp. 378–400, 2001.

[22] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," in *Proc. IEEE Int. Conf. Robotics and Automation*, San Francisco, CA, Apr. 1986, pp. 1419–1424.

[23] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *Int. J. Robot. Res.*, vol. 21, pp. 233–255, 2002.

[24] K. Kant and S. W. Zucker, "Planning collision free trajectories in time-varying environments: A two-level hierarchy," in *Proc. IEEE Int. Conf. Robotics and Automation*, Raleigh, NC, 1988, pp. 1644–1649.

[25] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, pp. 760–772, 1998.

[26] Z. Shiller, F. Large, and S. Sekhavat, "Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories," in *Proc. IEEE Int. Conf. Robotics and Automation*, Seoul, Korea, May 2001, pp. 3716–3721.

Zhihua Qu (M'90–SM'93) received the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, in 1990.

Since then, he has been with the University of Central Florida, Orlando. Currently, he is a Professor with the Department of Electrical and Computer Engineering. His main research interests are nonlinear systems, robust control and other advanced control techniques, robotics, and power systems. He has published a number of papers in these areas and is the author of two books, *Robust Control of Nonlinear Uncertain Systems* (New York: Wiley Interscience, 1998) and *Robust Tracking Control of Robotic Manipulators* (Piscataway, NJ: IEEE Press, 1996). He is presently serving as an Associate Editor for *Automatica* and for the *International Journal of Robotics and Automation*.

Jing Wang (M'01) received the B.Eng. and Ph.D. degrees in control theory and control engineering, both from Central South University of Technology, Changshu, China, in 1992 and 1997, respectively.

He was a Postdoctoral Research Fellow with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, from 1997 to 1999, and with the National University of Singapore, Singapore, from 1999 to 2002. He visited the Hong Kong Polytechnic University as a Research Associate in 1998. Since March 2002, he has been with School of Electrical and Computer Science, University of Central Florida, Orlando, as a Postdoctoral Visiting Scholar. His current research interests include adaptive nonlinear control, neural networks and fuzzy control, robot control and motion planning, trajectory optimization, and control applications.

Dr. Wang is a member of the AIAA. He was the co-recipient of the Best Theoretical Paper Award in 2002 at the Fourth World Congress on Intelligent Control and Automation, Shanghai, China.

Clinton E. Plaisted (M'02) received the B.S. degree in aerospace engineering from the Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, in 1999. He is currently working toward the M.S. degree in aerospace engineering at the University of Central Florida, Orlando.

In 1999, he joined Lockheed Martin Missiles and Fire Control, Orlando, FL. As a Senior Research Engineer, his main research interests are robust control and other advanced control techniques, missile flight mechanics, fire control systems, and robotics. He has published a number of conference papers in these areas. He is presently a Flight Controls Engineer at A.I. Solutions, Inc., Expendable Launch Vehicles, Mission Analysis Branch for NASA, Kennedy Space Center, Orlando, FL.