Auton Robot (2012) 32:207–226
DOI 10.1007/s10514-011-9267-7

Distributed reactive collision avoidance

Emmett Lalish · Kristi A. Morgansen

Received: 15 January 2011 / Accepted: 30 November 2011 / Published online: 21 January 2012

Abstract The work contained in this paper concerns a novel approach to the n-vehicle collision avoidance problem. The vehicle model used here allows for three-dimensional movement and represents a wide range of vehicles. The algorithm works in conjunction with any desired controller to guarantee all vehicles remain free of collisions while attempting to follow their desired control. This algorithm is reactive and distributed, making it well suited for real time applications, and explicitly accounts for actuation limits. A robustness analysis is presented which provides a means to account for delays and unmodeled dynamics. Robustness to an adversarial vehicle is also presented. Results are demonstrated in simulation.

Keywords Collision avoidance · Deconfliction · Safety

1 Introduction

As multi-vehicle autonomous systems are studied and implemented, the issue of conflict resolution becomes increasingly important. From mobile robots performing a cooperative search to air traffic control for unmanned aerial vehicles, collision avoidance is of utmost importance for safety. Much of the work so far on collision avoidance has been sponsored by the FAA to support a potential move to free-flight air traffic control, whereby aircraft can avoid each other in a decentralized manner rather than relying on a land-based controller. Similar concepts have been discussed regarding autonomous harbor control for ships. These scenarios will become more important as unmanned vehicles are introduced, because safety will need to be guaranteed for them to be accepted by their manned counterparts.

Currently, the FAA is supporting research into Sense and Avoid, the aim of which is to endow unmanned aerial vehicles with situational awareness and prescribed maneuvers that yield equivalent safety to a human pilot, as stated in Eurocontrol (2007). While this approach makes sense in the near term, eventually a method with provable safety may be desired, since even human pilots occasionally (though rarely) cause collisions. The work presented in this paper is focused on proving safety, and thus it ignores the unpredictable human element. However, noting the strengths and weaknesses of the guarantees herein may give insight into what parts of this problem are easily solved by a computer and what parts may benefit from being combined with human decision making.

A wide range of methods have addressed the collision avoidance problem, and in their survey paper, Kuchar and Yang (2000) split those methods into three categories: prescribed, optimized, and force field. In prescribed maneuver approaches (Hwang 2002; Stanley 2005; Pallottino et al. 2007), all vehicles follow a set protocol, not unlike the rules of the road. These approaches tend to yield a discrete event controller which, when combined with the vehicle's continuous dynamics, forms a hybrid system. Optimization methods (Fiorini and Shiller 1998; Frazzoli et al. 2001; Pallottino et al. 2002; Pongpunwattana and Rysdyk 2004; Hill et al. 2005; Guy et al. 2009) attempt to find the best route for all the vehicles to take to avoid each other while

E. Lalish (✉)
Moiré Inc., 1180 NW Maple St., Issaquah, WA 98027, USA
e-mail: emmett@moireinc.com

K.A. Morgansen
Department of Aeronautics and Astronautics, University of Washington, Box 352400, Seattle, WA 98195-2400, USA
e-mail: morgansen@aa.washington.edu

minimizing a cost function. Generally, these methods use a look-ahead or time horizon so that the solution does not have to be recalculated often. Force field approaches (Eby and Kelly 1999; Lalish et al. 2006; Mastellone et al. 2008; Wang and Schaub 2008; Roussos et al. 2008; Hoffmann and Tomlin 2008) use continuous feedback mechanisms to compute the control. Commonly, the force field between two vehicles is akin to the repulsion between two like-charged particles, however, many possible alternatives are available for feedback schemes. These approaches are generally reactive in that the control reacts to the current state of the system, rather than planning a trajectory ahead of time.

The collision cone concept, introduced by Chakravarthy and Ghose (1998) and subsequently used in the deconfliction literature (Fiorini and Shiller 1998; Frazzoli et al. 2001; Carbone et al. 2006; Fasano et al. 2008), is a first order look ahead for detecting conflicts. The collision cone (or velocity obstacle) is a set of velocities for one vehicle that will cause it to collide with another, assuming each of their velocities are constant. While most algorithms using the collision cone define collision by the distance between two points (as though each vehicle is a disk or sphere), Chakravarthy and Ghose (1998) shows how the method can be extended to irregularly shaped objects. Note that in order to handle walls, mazes, and other cluttered environments, the collision cone must be truncated by a finite look-ahead, as in Guy et al. (2009). In order to make stronger safety guarantees, the approach presented here does not truncate the collision cone, and hence applies only to unbounded environments.

An important differentiator between collision avoidance algorithms is their degree of centralization. Centralized means that all of the information about the vehicles is sent to a single server where the controls are calculated for each vehicle and sent back. The current air traffic control system uses centralization (with human operators), as do most optimization schemes (Frazzoli et al. 2001; Pallottino et al. 2002; Pongpunwattana and Rysdyk 2004). The other extreme is decentralization, in which no central server exists, and each vehicle computes its own control based only on other vehicles within a particular range of itself (Hill et al. 2005; Pallottino et al. 2007; Hoffmann and Tomlin 2008). This situation is ideal from a computational perspective because the number of vehicles that can be within range at a given time is bounded, and therefore so is the time the algorithm will take to run, regardless of the total number of vehicles involved. In between the two extremes is a category called distributed, in which no central server is used, and each vehicle computes its own control, but a vehicle may need more than local information to do so. In this case, computations scale with the number of vehicles involved (n), but the scaling is at least $O(n)$ better than the centralized case because now n separate processors are computing in parallel rather than one server doing all the work.

The vehicle model used in each formulation is also of utmost importance to the algorithm's applicability to real systems. First, for collision avoidance to be nontrivial, the dynamics must be at least second order (acceleration-level control input), and the control inputs must be bounded to model the difficulties of overcoming inertia. Many vehicles (cars, ships, airplanes, etc.) have nonholonomic constraints, and as such are often modeled at a high level by unicycle dynamics. The inputs to this model are forward acceleration and heading rate (which is equivalent to acceleration normal to the velocity). All vehicles have a maximum speed, but some vehicles (notably airplanes) also have a positive minimum speed. The constant-speed unicycle is one of the most broadly applicable vehicle models because it is the most constrained (less constrained vehicles can still duplicate its movements, but not vice-versa). Therefore an algorithm does well to apply to a constant-speed unicycle, but it is also helpful to make use of extra vehicle capabilities when available.

Many collision avoidance algorithms are limited in the number of vehicles for which they can guarantee avoidance. Some work for an arbitrary number (Pallottino et al. 2002, 2007; Frazzoli et al. 2001), while others are only guaranteed for two (Leitmann and Skowronski 1977; Carbone et al. 2006; Wang and Schaub 2008) or three (Hoffmann and Tomlin 2008) vehicles. Many more do not guarantee anything but instead provide a heuristic algorithm that attempts to deconflict the vehicles (Hill et al. 2005; Eby and Kelly 1999; Lalish et al. 2006).

Another important aspect of a collision avoidance algorithm is whether it works for a heterogeneous group of vehicles. Heterogeneity can be due to differences in vehicle size, dynamics, speed range, control authority, etc. Many algorithms are developed for a homogeneous group of vehicles for convenience and ease of notation, but could be easily extended to a heterogeneous case (Hill et al. 2005; Hoffmann and Tomlin 2008). Others, especially constant-speed algorithms, can have difficultly generalizing from a homogeneous state (Lalish et al. 2008; Pallottino et al. 2007). Another important consideration is that vehicle size, maximum speed, and control authority are often important parameters for the other vehicles in the system to know in order to properly avoid each other. In a homogeneous system, these parameters are known implicitly because they are the same for every vehicle, but in a heterogeneous system, those parameters must be exchanged. Communication is the most obvious tool, though sensors could be employed to identify the vehicle type and compare it to some known list, the same way a captain can identify a sailboat or a ferry and infer their size and maneuverability. Until this technology is fully mature, heterogeneity may require some degree of inter-vehicle communication.

A final term important to collision avoidance is liveness, which denotes the ability of the vehicles to attain their goals.

Liveness is important to consider because one simple way to avoid collisions is to have everyone stop moving. While this method may avoid collisions, it is not useful because the vehicles cannot arrive at their destinations. This type of situation is called a deadlock. Another problem scenario is a livelock, where the vehicles continue to move, but in such a way that they cannot reach their goals.

The Distributed Reactive Collision Avoidance (DRCA) algorithm presented here fills an important gap in the current collision avoidance literature. Namely, this algorithm presents a distributed, computationally efficient method to guarantee collision avoidance between n vehicles in the presence of arbitrary control authority limitations. No homogeneity is required among the vehicles that make up the system. The DRCA algorithm allows the vehicles to perform completely different tasks and to have different size, speed, actuation limitations, and gains. Plus, liveness of the solutions can be guaranteed, as well as robustness to modeling errors and adversarial behavior.

The DRCA algorithm is a two-step process, consisting of an optimization-based deconfliction maneuver, followed by the longer-term deconfliction maintenance phase, which is a reactive, force-field type approach. Both of the steps are based upon the collision cone concept. The optimization schemes are uniquely distributed, scale well computationally, and guarantee the existence of a feasible solution under given restrictions on initial conditions. The deconfliction maintenance controller builds upon previous work (Lalish et al. 2008; Lalish and Morgansen 2008) and ensures the vehicles follow their arbitrary outer-loop controllers as much as possible without sacrificing safety. This framework also allows vehicles to be given different priorities, such that lower priority vehicles will make way for higher priority ones.

This algorithm requires little (though nonzero) communication between vehicles and can safely add new vehicles to the system as they reach sensor range. Bounds are given on the required initial separation of vehicles, which implies one serious caveat: vehicles can only be added singly; merging groups does not yet have a safety guarantee. While the algorithm is distributed, it is not quite decentralized, in that all-to-all information is needed within a group. The information exchange can be easily implemented through a broadcast, and if enough bandwidth is available for vehicles to rebroadcast the information they receive, then a quasi-all-to-all topology can be achieved. The robustness analysis accounts for the corresponding time delays. Additionally, the algorithm can be run in a completely decentralized fashion, and while some of the guarantees no longer apply, simulations are given that demonstrate significant capability nonetheless.

The remainder of the paper is organized as follows. The problem statement is given in Sect. 2, including the vehicle model and definitions of conflict and collision. The DRCA algorithm is described in Sect. 3, which forms the bulk of the paper. Analyses of liveness and robustness are given in Sect. 4. Finally, simulation results are presented in Sect. 5, and concluding remarks are given in Sect. 6.

2 Problem statement

The work here presents a method for deconflicting n vehicles. Each vehicle has a nominal desired control input, $u_d(t)$, which comes from an arbitrary outer-loop controller. This controller is designed for the vehicle to perform a desired task, which could range from target tracking to waypoint navigation, area searching, etc. The goal of the DRCA algorithm is to adjust the control input on each vehicle to guarantee collision avoidance while simultaneously staying close to the desired control input (keeping in mind that this desired control can change with time).

2.1 Vehicle model

For this approach to collision avoidance, the only vehicle states that matter are position and velocity. Orientations affect performance, as they often have bearing on the magnitude of acceleration available in a particular direction, but they do not affect the underlying concepts of conflict and collision. In this way, many different vehicle models work equivalently with this approach.

The vehicle model chosen for the formulation of this problem is the generic 3D double-integrator. Specific vehicles are then modeled by restricting the control authority in particular directions by time- or state-dependent functions rather than by adjusting the dynamics themselves. In this way, this single model can represent two-dimensional systems or even nonholonomic vehicles such as constant-speed unicycles. Likewise, static obstacles can be avoided since they can be represented as vehicles with zero speed and zero control authority. Higher order dynamics (like an airplane banking to turn) can be accounted for with the robustness analysis.

The 3D double integrator dynamics are herein represented by

$$\frac{d}{dt}\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix}=\begin{bmatrix}\mathbf{v}\\ \mathbf{u}\end{bmatrix} \quad \text{and} \quad \frac{d}{dt}\Theta=\Omega\Theta, \tag{1}$$

where $\mathbf{r}, \mathbf{v}, \mathbf{u}\in\mathbb{R}^3$ are the position, velocity, and control input, respectively. The matrix $\Theta=[\hat{\mathbf{t}}, \hat{\mathbf{n}}, \hat{\mathbf{b}}]$ defines the orientation, and Ω is the cross product matrix of the body rotation vector $\boldsymbol{\omega}=[\omega_t, \omega_n, \omega_b]^{\mathsf{T}}$. The notation throughout this paper will use bold face for vectors, hats over unit-vectors, script capital letters for sets, standard capital letters for matrices and functions, and everything else is assumed scalar. Quantities subscripted with t, n, or b refer to the tangent,

normal, or binormal direction, respectively, while i and j subscripts reference the ith or jth vehicle.

Note that the orientation (defined by the $\hat{\mathbf{t}}$, $\hat{\mathbf{n}}$, and $\hat{\mathbf{b}}$ vectors) is only used as a local coordinate frame for the DRCA algorithm. The orientation does not directly affect the dynamics ($\mathbf{r}$ and $\mathbf{v}$) and, as such, can be arbitrary. However, many vehicle's input constraints are related to their orientation, and connecting this local coordinate frame to the actual body coordinates of the vehicle can be useful.

We constrain the input by use of an arbitrarily varying constraint set, $\mathbf{u}_i \in \mathscr{C}_i$. The only requirement is that $\mathscr{C}_i$ always contain the origin. Note this condition implies the position space must be unbounded, as maintaining velocity must always be allowed. A simple example of an input constraint set that limits maximum acceleration and velocity is

$$\mathscr{C}_i = \Big\{ \mathbf{u}_i \in \mathbb{R}^3 \big| \, \|\mathbf{u}_i\| \le u_{max}, \quad \|\mathbf{v}_i\| \ge v_{max} \implies \mathbf{u}_i^{\mathsf{T}} \mathbf{v}_i \le 0 \Big\}. \tag{2}$$

For the DRCA algorithm, one must choose a set of rectangular constraints $\mathscr{R}_i$ (which can also vary with time, state, etc.) for each vehicle that encloses its $\mathscr{C}_i$, as well as a corresponding saturation function, $S_i : \mathscr{R}_i \to \mathscr{C}_i$. The function S_i must be continuous, must become the identity map for any $\mathbf{u}_i \in \mathscr{C}_i$, and must preserve the sign of each component of $\mathbf{u}_i$ when decomposed in the $\hat{\mathbf{t}}$, $\hat{\mathbf{n}}$, and $\hat{\mathbf{b}}$ directions. In this example, one can choose

$$\mathscr{R}_i = \Big\{ \mathbf{u}_i \in \mathbb{R}^3 \big| \, -u_{max_i} \le u_{t_i} \le u_{max_i}, \dots \Big\}, \tag{3}$$

and

$$S_i = \begin{cases} \mathbf{u}_i \frac{u_{max}}{\|\mathbf{u}_i\|}, & \|\mathbf{u}_i\| > u_{max}, \\ \mathbf{u}_i - \frac{\mathbf{v}_i \mathbf{u}_i^{\mathsf{T}} \mathbf{v}_i}{v_{max}}, & \|\mathbf{v}_i\| \ge v_{max}, \mathbf{u}_i^{\mathsf{T}} \mathbf{v}_i \ge 0, \\ \mathbf{u}_i, & \text{otherwise.} \end{cases} \tag{4}$$

An example of how more complex vehicle dynamics can be represented by this simple model with an appropriate choice of input constraint set follows.

Example 1 Let us model a vehicle which can move forward with variable speed and turn in two axes (a 3D unicycle model) and with limits on its turn rate, forward acceleration, and maximum speed. One way to describe the model mathematically is by

$$\frac{d}{dt}\begin{bmatrix} \mathbf{r} \\ s \end{bmatrix} = \begin{bmatrix} s\hat{\mathbf{t}} \\ u_a \end{bmatrix} \quad \text{and} \quad \frac{d}{dt}\Theta = \Omega\Theta,$$

where $|u_a| \le u_{a_{max}}$, $|\omega_n| \le \omega_{n_{max}}$, $|\omega_b| \le \omega_{b_{max}}$, and $|s| \ge s_{max} \implies u_a s \le 0$. Alternatively, an equivalent representation of the system is (1) with $\mathbf{u} = u_a\hat{\mathbf{t}} + \|\mathbf{v}\|\,\omega_b\hat{\mathbf{n}} - \|\mathbf{v}\|\,\omega_n\hat{\mathbf{b}}$. The tangent vector must be initialized to the same direction as the velocity vector, but the dynamics will keep them aligned from then on. In this case, $\mathscr{R}$ can be defined by

$$u_{t_{max}} = -u_{t_{min}} = u_{a_{max}},$$
$$u_{n_{max}} = -u_{n_{min}} = \|\mathbf{v}\|\,\omega_{b_{max}},$$
$$u_{b_{max}} = -u_{b_{min}} = \|\mathbf{v}\|\,\omega_{n_{max}},$$

and the accompanying saturation function is

$$S = \begin{cases} \mathbf{u} - \frac{\mathbf{v}\mathbf{u}^{\mathsf{T}}\mathbf{v}}{s_{max}}, & \|\mathbf{v}\| \ge s_{max}, \mathbf{u}^{\mathsf{T}}\mathbf{v} \ge 0, \\ \mathbf{u}, & \text{otherwise.} \end{cases}$$

Normally, one would not equate the dynamics of a holonomic model to those of a nonholonomic one, largely because of differences in controllability. However, the DRCA algorithm is not dependent on controllability the way traditional control systems are.

The relative position vector from vehicle i to vehicle j is denoted $\tilde{\mathbf{r}}_{ij} \equiv \mathbf{r}_j - \mathbf{r}_i$, while the relative velocity vector is defined in the opposite sense: $\tilde{\mathbf{v}}_{ij} \equiv \mathbf{v}_i - \mathbf{v}_j$. Note that these definitions imply that $\dot{\tilde{\mathbf{r}}}_{ij} = -\tilde{\mathbf{v}}_{ij}$, and $\dot{\tilde{\mathbf{v}}}_{ij} = \mathbf{u}_i - \mathbf{u}_j$.

To compare different systems in which collision avoidance is to be implemented, a dimensionless Deconfliction Difficulty Factor, η, will be of use. This factor is defined as

$$\eta = \frac{v_{max}^2}{u_{max} d_{sep}}. \tag{5}$$

Conceptually, this factor is the ratio of the worst case turning radius to the required separation distance, d_{sep}. It can also be thought of in terms of stopping distance.

The DRCA algorithm was designed primarily for systems with large η (greater than unity) such as aircraft and ships, where the collision avoidance task is difficult because of the dominance of vehicle inertia. Vehicles with small η (significantly less than unity) such as mobile robots can often be modeled as having a direct velocity command, since the inertia becomes insignificant. In such cases, potential function methods for collision avoidance may be preferable to the DRCA algorithm because of their simplicity and ability to closely pack vehicles. The downside of potential function methods is their general lack of guarantees, especially when inertia is considered.

2.2 Conflicts and collisions

The vehicles considered here are modeled as point masses, however, physical vehicles have finite size. Therefore, to account for physical constraints in the theoretical model, the condition for collision is not to attain the same position in space at the same time but rather to come within a minimum allowed distance at some point in time. This minimum dis-

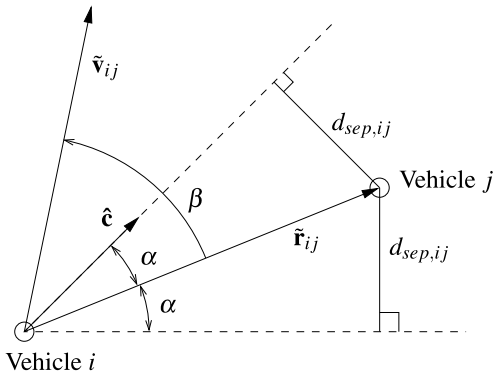

Fig. 1 A 2D section of the collision cone along the $\tilde{\mathbf{r}}_{ij}$-$\tilde{\mathbf{v}}_{ij}$ plane. The area between the two dotted lines is the collision cone; a conflict occurs when the relative velocity vector, $\tilde{\mathbf{v}}_{ij}$, lies within this area. Technically, position and velocity vectors should not be plotted together since they have different units, but an exception is made here to show the relationship between their angles

tance could be, for example, the five nautical mile separation between aircraft required by the FAA or the sum of the radii of two vehicles.

Definition 1 (Collision) A collision occurs between vehicles i and j when $\|\tilde{\mathbf{r}}_{ij}\| < d_{sep,ij}$, where $d_{sep,ij}$ is the minimum allowed separation distance between the vehicles' geometric centers.

For two vehicles not actively in a collision, the next question is whether they will collide if they remain on their present course and speed. This situation will be called a conflict.

Definition 2 (Conflict) A conflict occurs between vehicles i and j if they are not currently in a collision, but with zero control input (i.e. constant velocity), at some future point in time they will enter a collision:

$$\min_{t>0} \left\| \tilde{\mathbf{r}}_{ij} - t\tilde{\mathbf{v}}_{ij} \right\| < d_{sep,ij}. \tag{6}$$

The following lemma provides a useful way to check for conflicts. To simplify the notation in the rest of this paper, the ij subscripts will generally be suppressed (for example, $\tilde{\mathbf{r}}_{ij}$ will be written as $\tilde{\mathbf{r}}$).

Lemma 1 *Let* $\beta = \angle\tilde{\mathbf{v}} - \angle\tilde{\mathbf{r}}$ *and* $\alpha = \arcsin(\frac{d_{sep}}{\|\tilde{\mathbf{r}}\|})$. *A necessary and sufficient condition for no conflict to occur is* $|\beta| \geq \alpha$.

The angle α represents the half-width of the collision cone (Chakravarthy and Ghose 1998; Frazzoli et al. 2001; Carbone et al. 2006), which is depicted in Fig. 1. A proof using this notation is given by Lalish and Morgansen (2008), but it is conceptually the same as the original collision cone proofs from Chakravarthy and Ghose (1998).

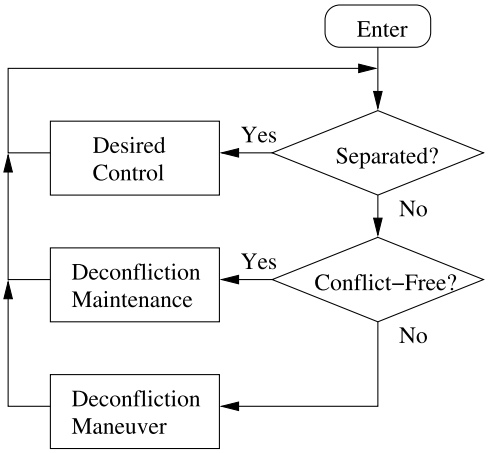

Fig. 2 Flow chart of the DRCA Algorithm

3 DRCA

The DRCA algorithm uses a two-step process: a deconfliction maneuver and a deconfliction maintenance phase. The guarantees of n-vehicle collision avoidance rest upon three theorems. The first is that if a group of vehicles is conflict-free, the deconfliction maintenance controller will keep them that way. If a vehicle starts in conflict with one or more other vehicles, the second theorem states it can perform a deconfliction maneuver quickly enough to ensure no collisions happen during the maneuver. The third theorem gives a conservative separation criterion to ensure the deconfliction maneuver will bring the vehicle to a conflict-free state.

The basic algorithm a vehicle uses is to first check the separation criterion to see if the other vehicles are close enough to worry about. If so, conflict is checked. If a conflict is found, a maneuver is performed until a conflict-free state is reached. Once conflict-free, the deconfliction maintenance controller is used to keep them that way. These steps yield the flow chart for the DRCA algorithm that is shown in Fig. 2. The deconfliction maintenance phase also takes the desired control into account, though there is no guarantee that it is followed at all times.

The first theorem and proof, along with a description of the deconfliction maintenance controller, are presented in Sect. 3.1. The maneuver is described in Sect. 3.2 along with the second two theorems and proofs. The pieces are brought together in Sect. 3.3, which describes the algorithm in detail and the conditions and caveats implicit in these collision avoidance guarantees. Section 3.4 shows how the algorithm can be adjusted to allow for cases when the conditions for the guarantees are not met. This heuristic backup performs well at avoiding collisions, as shown in Sect. 5, meaning the guarantees are indeed conservative, and the algorithm has some robustness around the sufficient conditions.

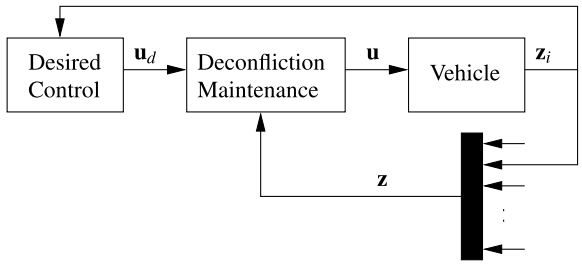

Fig. 3 Block diagram of the system when using the deconfliction maintenance controller. The vector $\mathbf{z} = [\mathbf{z}_1^\mathsf{T}, \mathbf{z}_2^\mathsf{T}, \ldots, \mathbf{z}_n^\mathsf{T}]^\mathsf{T}$ and $\mathbf{z}_i = [\mathbf{r}_i^\mathsf{T}, \mathbf{v}_i^\mathsf{T}]^\mathsf{T}$. Note that the deconfliction maintenance block acts as a type of saturation on the desired control, $\mathbf{u}_d$

3.1 Deconfliction maintenance

The deconfliction maintenance controller is used to keep the system of vehicles conflict-free. This controller allows each vehicle to use its desired control input unless that input would cause the vehicle to come into conflict with another vehicle. A basic block diagram of this control system is shown in Fig. 3.

In order to smoothly transition between the desired control and the avoidance control, each vehicle needs a way to measure how close its velocity vector is to causing a conflict. The first step is to construct a unit-vector, $\hat{\mathbf{c}}$, representing the side of the collision cone nearest $\tilde{\mathbf{v}}$. The vector $\hat{\mathbf{c}}$ is found by rotating $\tilde{\mathbf{r}}$ by α around a vector $\mathbf{q} = \tilde{\mathbf{r}} \times \tilde{\mathbf{v}}$ and normalizing:

$$\hat{\mathbf{c}} = \frac{\tilde{\mathbf{r}}}{\|\tilde{\mathbf{r}}\|}\cos\alpha + \left(\frac{\mathbf{q}\times\tilde{\mathbf{r}}}{\|\mathbf{q}\|\|\tilde{\mathbf{r}}\|}\right)\sin\alpha. \tag{7}$$

Next, construct a normal vector, $\mathbf{e}$, from the collision cone to the relative velocity vector, $\tilde{\mathbf{v}}$ (see Fig. 4). If $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$, then $\mathbf{e} \equiv (I - \hat{\mathbf{c}}\hat{\mathbf{c}}^\mathsf{T})\tilde{\mathbf{v}}$, but if $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} \leq 0$ (the vehicles are headed away from each other), then no normal exists, and the nearest point on the collision cone is the tip, so $\mathbf{e} \equiv \tilde{\mathbf{v}}$. Therefore:

$$\mathbf{e} \equiv \begin{cases} \tilde{\mathbf{v}}, & \hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} \leq 0, \\ (I - \hat{\mathbf{c}}\hat{\mathbf{c}}^\mathsf{T})\tilde{\mathbf{v}}, & \hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0. \end{cases} \tag{8}$$

In order to combine the effects of multiple collision cones, the system is decomposed into three component directions, and those directions will be analyzed separately. Let the coordinate system be defined by the orthonormal vectors $\hat{\mathbf{t}}$, $\hat{\mathbf{n}}$, and $\hat{\mathbf{b}}$. The orientation of this coordinate system is arbitrary, but the convention of using tangent, normal, and binormal notation is chosen since fixing the coordinates to the body of the vehicle often simplifies analysis.

The next step is to determine how much control (change in velocity) can be applied in each of these directions before a conflict forms. For simplicity, a conservative approach is taken whereby the signed distance is found from $\tilde{\mathbf{v}}$ to the tangent plane enclosing the collision cone (defined by the normal vector $\mathbf{e}$) in each of the $\hat{\mathbf{t}}$, $\hat{\mathbf{n}}$, and $\hat{\mathbf{b}}$ directions. These signed distances are

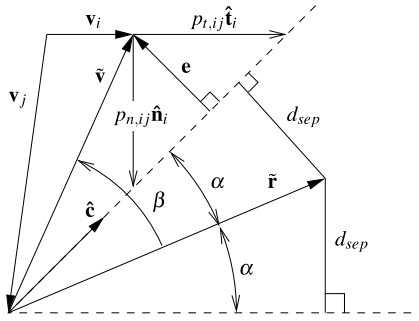

Fig. 4 Geometry of the $\mathbf{e}$ and $\hat{\mathbf{c}}$ vectors, as seen in an $\tilde{\mathbf{r}}$-$\tilde{\mathbf{v}}$ section through the 3D collision cone (*dotted lines*). The conflict measures p_t and p_n are shown, but $\hat{\mathbf{b}}$ points into the page, so p_b is infinite. In this example, $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$

$$p_{t,ij} = \frac{\|\mathbf{e}_{ij}\|^2}{\mathbf{e}_{ij}^\mathsf{T}\hat{\mathbf{t}}_i}, \qquad p_{n,ij} = \frac{\|\mathbf{e}_{ij}\|^2}{\mathbf{e}_{ij}^\mathsf{T}\hat{\mathbf{n}}_i}, \quad \text{and}$$

$$p_{b,ij} = \frac{\|\mathbf{e}_{ij}\|^2}{\mathbf{e}_{ij}^\mathsf{T}\hat{\mathbf{b}}_i},$$

which are depicted in Fig. 4.

Select $\varepsilon_t, \varepsilon_n, \varepsilon_b > 0$ as thresholds such that when $|p_t| > \varepsilon_t$, the conflict is far enough away that it can be ignored (and likewise for p_n and p_b). The n-vehicle deconfliction maintenance controller running on vehicle i computes p_t, p_n, and p_b to each of the other vehicles and then finds the closest conflict in each direction, i.e.,

$$\begin{aligned} p_{t_i}^+ &= \min_j \left\{ p_{t,ij} > 0, \varepsilon_{t_i} \right\}, \\ p_{t_i}^- &= -\max_j \left\{ p_{t,ij} < 0, -\varepsilon_{t_i} \right\}, \end{aligned} \tag{9}$$

and likewise for p_n and p_b. Note that by definition $0 < p^\pm \leq \varepsilon$. To simplify notation, in any case where a relation holds in all of the tangent, normal, and binormal directions, the subscript will be suppressed.

The input is constructed using a function, F, such that in each direction $u = F(p^+, p^-)$. The control function chosen for this implementation of the DRCA algorithm is

$$\begin{aligned} F(p^+, p^-) = {} & \frac{u_{min}}{\varepsilon}p^+ + \frac{u_{max}}{\varepsilon}p^- \\ & + \frac{u_d - u_{max} - u_{min}}{\varepsilon^2}p^+p^-, \end{aligned} \tag{10}$$

because it is a bilinear interpolation of the following ordered triples of the form (p^+, p^-, u):

$$P_1 = (0, 0, 0) \qquad P_2 = (\varepsilon, 0, u_{min}),$$

$$P_3 = (0, \varepsilon, u_{max}) \qquad P_4 = (\varepsilon, \varepsilon, u_d).$$

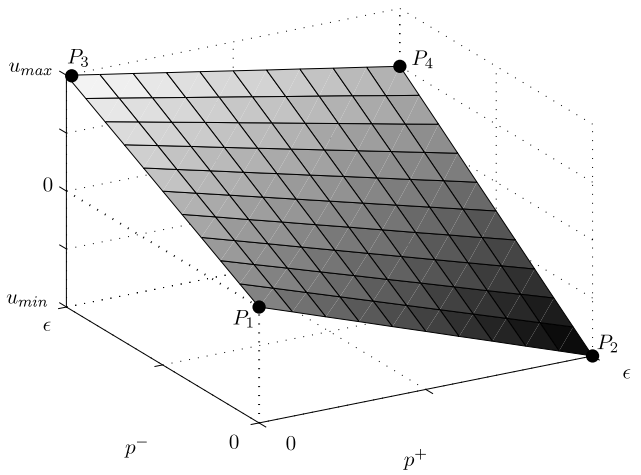

Fig. 5 Example of the control function, F. Note that P_4 moves up and down with changing u_d

An example of this control function is shown in Fig. 5. Because F depends on the desired control, u_d must be saturated such that

$$u_{min} \leq u_d \leq u_{max}. \tag{11}$$

This choice of control function means that once the $\mathbf{u}$ vector is constructed from its three components, then $\mathbf{u} \in \mathscr{R}$.

Let the gain, k, be defined as the maximum gradient of the function F (for any u_d). For (10), the following holds:

$$k = \frac{u_{max} - u_{min}}{\varepsilon}. \tag{12}$$

Note that k has units of inverse seconds, and implicitly, k_t, k_n, and k_b can have different values. These gains will be used in further analyses, since they are more generic than ε.

Assuming the system consists of n vehicles, this algorithm's computation time on each vehicle scales as $O(n)$, since it only requires the computation of each other vehicle's collision cone and then substitution of these results into the control function. Technically $O(n^2)$ computations occur in the entire group, but since these computations are independent of each other (only linked by the sensed or communicated states), they can happen in parallel in a distributed fashion, so only the per-vehicle scaling matters.

Theorem 1 *The deconfliction maintenance controller described above, when implemented on n vehicles with dynamics* (1) *and inputs constrained by* $\mathscr{C}_i$*, will keep the system collision free for all time if the system starts conflict-free.*

Proof To measure the distance to a collision, define m as a signed version of $\|\mathbf{e}\|$ (in terms of $\tilde{\mathbf{v}}$ from the geometry in Fig. 4):

$$m \equiv \begin{cases} \|\tilde{\mathbf{v}}\|, & \hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} \leq 0, \\ \|\tilde{\mathbf{v}}\| \sin(|\beta| - \alpha), & \hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0. \end{cases} \tag{13}$$

Note that m is negative during conflict and positive during non-conflict.

To ensure that a conflicted state is never reached (i.e. m is always greater than zero), it is sufficient to show that for every pair of vehicles there is a neighborhood on the positive side of $m = 0$ where $\dot{m} \geq 0$. This condition implies that as a velocity vector and the boundary of a collision cone approach each other, they will either stop approaching or recede before a conflict is formed. The fact that this neighborhood is one-sided is important because $\dot{m}$ does not exist at $m = 0$ for the same reason that $\frac{d}{dt}\|\tilde{\mathbf{v}}\|$ does not exist at $\tilde{\mathbf{v}} = 0$. However, if this condition is satisfied, then $m > 0$ for all time, ensuring that $m \neq 0$.

For $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} \leq 0$,

$$\dot{m} = \frac{\tilde{\mathbf{v}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}}{\|\tilde{\mathbf{v}}\|} = \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}, \tag{14}$$

where $\hat{\mathbf{e}} = \mathbf{e}/\|\mathbf{e}\|$. Expanding $\mathbf{u}_i$ into its components yields $\mathbf{u}_i = u_{t_i}\hat{\mathbf{t}}_i + u_{n_i}\hat{\mathbf{n}}_i + u_{b_i}\hat{\mathbf{b}}_i$. Next substitute for $\dot{\tilde{\mathbf{v}}}$ in (14) to get (using the ij notation again briefly for clarity)

$$\begin{aligned} \hat{\mathbf{e}}_{ij}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}_{ij} = {} & u_{t_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{t}}_i + u_{n_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{n}}_i + u_{b_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{b}}_i \\ & - u_{t_j}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{t}}_j - u_{n_j}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{n}}_j - u_{b_j}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{b}}_j. \end{aligned}$$

Because of the symmetry of the problem, $\mathbf{e}_{ji} = -\mathbf{e}_{ij}$ and $m_{ij} = m_{ji}$, so

$$\begin{aligned} \hat{\mathbf{e}}_{ij}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}_{ij} = {} & u_{t_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{t}}_i + u_{n_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{n}}_i + u_{b_i}\hat{\mathbf{e}}_{ij}^{\mathsf{T}}\hat{\mathbf{b}}_i \\ & + u_{t_j}\hat{\mathbf{e}}_{ji}^{\mathsf{T}}\hat{\mathbf{t}}_j + u_{n_j}\hat{\mathbf{e}}_{ji}^{\mathsf{T}}\hat{\mathbf{n}}_j + u_{b_j}\hat{\mathbf{e}}_{ji}^{\mathsf{T}}\hat{\mathbf{b}}_j \\ = {} & m_{ij}\left(\frac{u_{t_i}}{p_{t,ij}} + \frac{u_{n_i}}{p_{n,ij}} + \frac{u_{b_i}}{p_{b,ij}}\right. \\ & \left. + \frac{u_{t_j}}{p_{t,ji}} + \frac{u_{n_j}}{p_{n,ji}} + \frac{u_{b_j}}{p_{b,ji}}\right). \end{aligned} \tag{15}$$

For $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0$, the derivative of (13) becomes

$$\dot{m} = \sin(|\beta| - \alpha)\frac{d\|\tilde{\mathbf{v}}\|}{dt} + \|\tilde{\mathbf{v}}\|\cos(|\beta| - \alpha)\frac{d}{dt}(|\beta| - \alpha). \tag{16}$$

The derivative exists because $m > 0$ implies $\|\tilde{\mathbf{v}}\| \neq 0$, $|\beta| \geq \alpha > 0$, and $\|\tilde{\mathbf{r}}\| \geq d_{sep} > 0$. From the geometry,

$$\frac{d\,|\beta|}{dt} = \operatorname{sgn}(\beta)\left(\frac{d\angle\tilde{\mathbf{v}}}{dt} - \frac{d\angle\tilde{\mathbf{r}}}{dt}\right)$$
$$= \operatorname{sgn}(\beta)\frac{d\angle\tilde{\mathbf{v}}}{dt} + \frac{\|\tilde{\mathbf{v}}\|}{\|\tilde{\mathbf{r}}\|}|\sin\beta|.$$

The derivative of α is somewhat less straight-forward:

$$\frac{d\alpha}{dt} = \frac{d}{dt}\left(\arcsin\left(\frac{d_{sep}}{\|\tilde{\mathbf{r}}\|}\right)\right)$$
$$= \frac{d}{dt}\left(\frac{d_{sep}}{\|\tilde{\mathbf{r}}\|}\right)\left(1 - \left(\frac{d_{sep}}{\|\tilde{\mathbf{r}}\|}\right)^2\right)^{-1/2}$$
$$= \frac{d_{sep}\|\tilde{\mathbf{v}}\|\cos\beta}{\|\tilde{\mathbf{r}}\|^2}\left(\frac{\|\tilde{\mathbf{r}}\|}{\sqrt{\|\tilde{\mathbf{r}}\|^2 - d_{sep}^2}}\right)$$
$$= \frac{\|\tilde{\mathbf{v}}\|}{\|\tilde{\mathbf{r}}\|}\cos\beta\tan\alpha.$$

Combining the above two terms gives

$$\frac{d}{dt}(|\beta| - \alpha) = \frac{\|\tilde{\mathbf{v}}\|}{\|\tilde{\mathbf{r}}\|}(|\sin\beta| - \cos\beta\tan\alpha) + \operatorname{sgn}(\beta)\frac{d\angle\tilde{\mathbf{v}}}{dt},$$

which can be substituted into (16) to get

$$\dot{m} = \sin(|\beta| - \alpha)\frac{d\|\tilde{\mathbf{v}}\|}{dt} + \cos(|\beta| - \alpha)\operatorname{sgn}(\beta)\|\tilde{\mathbf{v}}\|\frac{d\angle\tilde{\mathbf{v}}}{dt}$$
$$+ \cos(|\beta| - \alpha)\frac{\|\tilde{\mathbf{v}}\|^2}{\|\tilde{\mathbf{r}}\|}(|\sin\beta| - \cos\beta\tan\alpha). \tag{17}$$

To simplify the above, note that in the case of $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0$, the geometry of the vectors (Fig. 4) gives

$$\hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}} = \sin(|\beta| - \alpha)\frac{d\|\tilde{\mathbf{v}}\|}{dt} + \cos(|\beta| - \alpha)\operatorname{sgn}(\beta)\|\tilde{\mathbf{v}}\|\frac{d\angle\tilde{\mathbf{v}}}{dt}.$$

Therefore (17) reduces to

$$\dot{m} = \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}} + \cos(|\beta| - \alpha)\frac{\|\tilde{\mathbf{v}}\|^2}{\|\tilde{\mathbf{r}}\|}(|\sin\beta| - \cos\beta\tan\alpha).$$

This expression can be further simplified by recognizing that

$$\frac{m}{\|\tilde{\mathbf{v}}\|\cos\alpha} = \frac{\sin|\beta|\cos\alpha - \sin\alpha\cos|\beta|}{\cos\alpha}$$
$$= |\sin\beta| - \cos\beta\tan\alpha.$$

Therefore

$$\dot{m} = \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}} + m\frac{\|\tilde{\mathbf{v}}\|\cos(|\beta| - \alpha)}{\|\tilde{\mathbf{r}}\|\cos\alpha}. \tag{18}$$

The second term is always positive because $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0$ implies that $\cos(|\beta| - \alpha) > 0$, and $\alpha \le \pi/2$ by definition. Combining this result with (14) implies that

$$\dot{m} \ge \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}$$

for any value of $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}}$. Recalling (15),

$$\dot{m}_{ij} \ge m_{ij}\left(\frac{u_{t_i}}{p_{t,ij}} + \frac{u_{n_i}}{p_{n,ij}} + \frac{u_{b_i}}{p_{b,ij}}\right.$$
$$\left. + \frac{u_{t_j}}{p_{t,ji}} + \frac{u_{n_j}}{p_{n,ji}} + \frac{u_{b_j}}{p_{b,ji}}\right). \tag{19}$$

As long as the controller ensures that u_{t_i} has the same sign as $p_{t,ij}$, etc. then $\dot{m} \ge 0$ for that pair of vehicles. Note that each vehicle need only calculate its control from its own point of view, and this rule will automatically cause the vehicles to cooperate in avoiding conflicts.

Combining this result with the definitions (9), any continuous control function that satisfies

$$\lim_{p_{t_i}^+ \to 0^+} u_{t_i} \ge 0, \qquad \lim_{p_{t_i}^- \to 0^+} u_{t_i} \le 0,$$
$$\lim_{p_{n_i}^+ \to 0^+} u_{n_i} \ge 0, \qquad \lim_{p_{n_i}^- \to 0^+} u_{n_i} \le 0, \tag{20}$$
$$\lim_{p_{b_i}^+ \to 0^+} u_{b_i} \ge 0, \qquad \lim_{p_{b_i}^- \to 0^+} u_{b_i} \le 0,$$

also ensures that a neighborhood exists on the positive side of $m = 0$ for which $\dot{m} \ge 0$, guaranteeing the system cannot enter a conflicted state.

The control function (10) satisfies (20), so the deconfliction maintenance controller will cause the n-vehicle system to remain conflict-free for all time, assuming it started that way. □

Note that this result holds for arbitrary (even time varying) $\mathbf{u}_d$, $\mathbf{u}_{min}$ and $\mathbf{u}_{max}$, so long as they satisfy (11) and $\mathscr{R}$ contains the origin at every instant. In addition, $\mathbf{u}$ can be further saturated using S in order, conform to the non-rectangular constraint set $\mathscr{C}$ without affecting the guarantee. The key for this saturation to work is that S must preserve the octant of $\mathbf{u}$ such that (20) is still satisfied.

3.2 Deconfliction maneuver

If the vehicles start in conflict, something must be done to bring them to a conflict-free state before the deconfliction maintenance controller can do its job. In this subsection, an analysis is shown for the case of a single vehicle being added to a group of n vehicles that are conflict-free with each other, but not with the new vehicle. This set of conflict-free vehicles will be denoted $\mathscr{D}$. Section 3.3 will extend this analysis to more general n-vehicle systems.

The purpose of the deconfliction maneuver is to bring a vehicle to a conflict-free state with a guarantee that no collisions will occur during this maneuver, given certain bounds on the separation. These derived bounds, while sufficient, are conservative. It can be easily shown that some

restrictions on the initial conditions are necessary for any type of collision avoidance to be possible between vehicles with limited acceleration, however determining those exact bounds on feasibility for an n-vehicle system remains an open problem.

The deconfliction maneuver described here only works for vehicles that are capable of stopping. A similar maneuver has also been developed for constant-speed vehicles (or vehicles with a minimum speed), which has been omitted for brevity, but can be found in Lalish (2009). Both deconfliction maneuvers are fundamentally 2D (native 3D maneuvers are a topic of current research). In order to use this maneuver in a 3D application, one must project the vehicle's positions and velocities onto a plane (usually horizontal), perform the calculations, then add back in the original vertical component to the resulting velocity. The guarantees of attaining conflict-freedom will still apply (though the solutions will be more conservative) and once deconfliction maintenance takes over, the 3D nature of the system will be accounted for in a less conservative way.

To perform its maneuver, vehicle i calculates an optimization to find a new velocity vector, $\mathbf{v}_i'$, which is free of conflict with the other vehicles in $\mathscr{D}$ and minimizes $\|\Delta\mathbf{v}_i\|$, where $\Delta\mathbf{v}_i = \mathbf{v}_i' - \mathbf{v}_i$. While this minimization is nonconvex in general (see Fig. 6), it can be solved relatively quickly because only a finite number of possible points must be checked. The algorithm for solving this optimization works as follows.

The allowable space for $\mathbf{v}_i'$ is a disk, centered at the origin, of radius $v_{i,max}$. The optimal solution must lie on a boundary because the zero-cost point is not feasible (otherwise there would be no conflicts). This optimal solution must either be a vertex between two collision cones, the nearest point on a single collision cone, or the vertex between one collision cone and the edge of the allowable space. These points are depicted in Fig. 6. This problem and solution are similar to Guy et al. (2009), which was shown to be computationally efficient for systems with a great number of vehicles.

The first step of the algorithm is to compute each of the nearest points on each collision cone (a right and a left solution exist for each cone). These points are found by first computing the unit-vector $\hat{\mathbf{c}}$, now using a 2D version of (7):

$$\hat{\mathbf{c}} = R(\pm\alpha)\frac{\tilde{\mathbf{r}}}{\|\tilde{\mathbf{r}}\|}, \tag{21}$$

where R is the 2×2 rotation matrix. The nearest points are now given by $\mathbf{v}_i' = \hat{\mathbf{c}}\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} + \mathbf{v}_j$. This list of points is checked to make sure each $\|\mathbf{v}_i'\| \le v_{i,max}$. If not, then $\mathbf{v}_i'$ is replaced with

$$\mathbf{v}_i' = \mathbf{v}_j - \hat{\mathbf{c}}\hat{\mathbf{c}}^\mathsf{T}\mathbf{v}_j \pm \hat{\mathbf{c}}\sqrt{(\hat{\mathbf{c}}^\mathsf{T}\mathbf{v}_j)^2 - \mathbf{v}_j^\mathsf{T}\mathbf{v}_j + v_{i,max}^2},$$

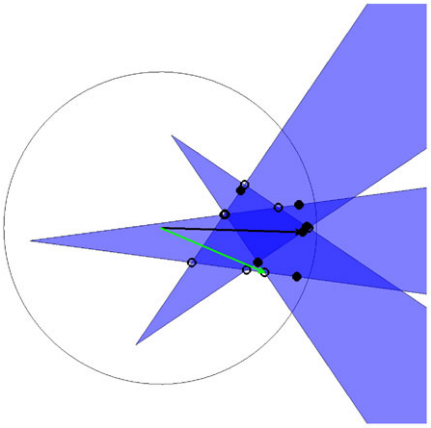

Fig. 6 Example optimization of a variable-speed maneuver. The *upper, darker vector* (*black* in the online color version) is $\mathbf{v}_i$, the *lower, lighter vector* (*green* in the online color version) is $\mathbf{v}_i'$, the *circle* is the edge of the allowable disk of solutions, and the *shaded regions* (*blue* in the online color version) are the collision cones. The tips of the collision cones are offset from the origin by $\mathbf{v}_j$ (this figure is like several copies of Fig. 4 overlaid). This plot is in the absolute velocity-space of vehicle i, so the vehicle positions cannot be directly represented any more than by the shape of the collision cones. The possible optima are also plotted; *solid circles* are one-cone points and open circles are two-cone points

where only the solution with the smaller $\|\Delta\mathbf{v}_i\|$ is kept. These points account for the only possible optima on the intersection of a cone and the edge of the space. Next, all of the two-cone intersection points are found using linear equations (keeping only those with $\|\mathbf{v}_i'\| \le v_{i,max}$). All of these points are ordered by increasing $\|\Delta\mathbf{v}_i\|$ and are checked consecutively for conflicts with the other vehicles. Because of the ordering, as soon as a point is found which is conflict free for all j, it is the optimal solution, and the algorithm terminates.

This optimization has a bound on its computation time since only a finite number of possible optima must be checked. Because some points are a combination of two cones, there are $O(n^2)$ points to check. Since these points need to be checked for conflict with the other $n-2$ collision cones, the maximum computation time is upper-bounded by cn^3, where c is related to the time each type of computation requires. This scaling is conservative because, in general, the algorithm will terminate long before it checks every point.

This analysis would guarantee a conflict-free solution if the vehicle could attain its desired velocity vector instantaneously. However, the limited control authority available makes this impossible. Instead, a finite amount of time is required for the vehicle to attain its desired velocity, and during that time it and the other vehicles move, causing the collision cones to move. In order to ensure that the system is still conflict-free after this motion, the initial collision cones must be enlarged to the point of enclosing all possible movements.

To bound the collision cone, one must simply bound $\|\Delta\tilde{\mathbf{r}}\| \leq \delta$, or how much the vehicles can change position before the maneuver is complete. Then the width of the collision cone is enlarged to

$$\alpha_e = \arcsin\left(\frac{d_{sep}+\delta}{\|\tilde{\mathbf{r}}\|}\right). \tag{22}$$

Note that this expression implies an initial separation of $\|\tilde{\mathbf{r}}\| \geq d_{sep}+\delta$, or else α_e will be undefined.

Lemma 2 *Consider two vehicles (i and j), each modeled by a planar version of (1). Vehicle j is subject to the maximum speed constraint $\|\mathbf{v}_j\| \leq v_{j,max}$. Vehicle i is subject to $\|\mathbf{u}_i\| \leq u_{i,max}$ and $\|\mathbf{v}_i\| \leq v_{i,max}$ and is moving with maximum acceleration from its initial velocity, $\mathbf{v}_i$, to its desired velocity, $\mathbf{v}_i'$. The relative motion between the vehicles in the time it takes vehicle i to attain its desired velocity is bounded by $\|\Delta\tilde{\mathbf{r}}\| \leq \delta$, where*

$$\delta = \frac{2v_{i,max}}{u_{i,max}}\left(v_{i,max}+v_{j,max}\right). \tag{23}$$

Proof The velocity change of the maneuver is bounded by $\|\Delta\tilde{\mathbf{v}}_i\| \leq 2v_{i,max}$, which means the time, t, for the maneuver to complete is bounded by

$$t \leq \frac{2v_{i,max}}{u_{i,max}}. \tag{24}$$

Let $\Delta\mathbf{r}_i$ denote the net motion of vehicle i during the maneuver, such that $\Delta\mathbf{r} = \Delta\mathbf{r}_j - \Delta\mathbf{r}_i$. Conservatively, $\|\Delta\mathbf{r}_i\| \leq v_{i,max}t$ and $\|\Delta\mathbf{r}_j\| \leq v_{j,max}t$. Therefore $\|\Delta\tilde{\mathbf{r}}\| \leq \delta$. □

This bound can in turn be used in (22) to size the enlarged collision cone. Note that for a homogeneous group of vehicles, this bound can be written in terms of the deconfliction difficulty factor as $\delta = 4\eta d_{sep}$. The following theorem states how this bound can be used to keep vehicles from colliding during a deconfliction maneuver.

Theorem 2 *Consider a set of vehicles, $\mathscr{D}$, which are not in conflict with each other and are preforming deconfliction maintenance. When another vehicle, i, is in conflict with some or all members of $\mathscr{D}$ and performs the deconfliction maneuver, the system will come to a conflict-free state and no vehicles will collide during the maneuver. The vehicles are all modeled by a planar version of (1), have speed constraints $\|\mathbf{v}_j\| \leq v_{j,max}$, and vehicle i has the input constraint $\|\mathbf{u}_i\| \leq u_{i,max}$. A feasible solution to the optimization problem, $\mathbf{v}_i'$, is assumed to exist, and the vehicles in $\mathscr{D}$ maintain a conflict-free state with $\mathbf{v}_i'$, using a cone with width defined by (22) and (23).*

Proof If the optimization problem is feasible, then the optimal solution is guaranteed to be found, and this point will satisfy

$$\left|\angle\tilde{\mathbf{v}}' - \angle\tilde{\mathbf{r}}\right| \geq \arcsin\left(\frac{d_{sep}+\delta}{\|\tilde{\mathbf{r}}\|}\right).$$

The maximum amount of time required for the maneuver is given by (24), and during this time $\|\Delta\tilde{\mathbf{r}}\| \leq \delta$ as given by Lemma 2. Therefore, once the desired velocities have been attained, one still has

$$\left|\angle\tilde{\mathbf{v}}' - \angle(\tilde{\mathbf{r}}+\Delta\tilde{\mathbf{r}})\right| \geq \arcsin\left(\frac{d_{sep}}{\|\tilde{\mathbf{r}}\|}\right),$$

meaning the vehicles are not in conflict. The vehicles cannot collide during this time because, as stated earlier, the pairs must be initially separated by at least $\|\tilde{\mathbf{r}}\| \geq d_{sep}+\delta$, which means after the maneuver, they still must be outside of collision because $\|\tilde{\mathbf{r}}+\Delta\tilde{\mathbf{r}}\| \geq d_{sep}$. □

Of course, all of this is for naught if a feasible optimization solution does not exist. The following theorem gives a conditional bound on initial separation that is sufficient to guarantee the existence of a solution.

Theorem 3 *For vehicle i of a planar n-vehicle system, let vehicle i's speed be constrained by $\|\mathbf{v}_i\| \leq v_{i,max}$, while each other vehicle's speed is constrained by the uniform bound $\|\mathbf{v}_j\| \leq v_{max}$. Let the separation criterion, σ_i, be defined as*

$$\sigma_i \equiv \sum_{j\in\mathscr{D}} \alpha_e, \tag{25}$$

and the separation bound be defined as

$$\rho_i \equiv \begin{cases} \arcsin(\frac{v_{i,max}}{v_{max}}), & v_{i,max} < v_{max}, \\ \frac{\pi}{2}, & v_{i,max} \geq v_{max}. \end{cases} \tag{26}$$

There exists an admissible velocity vector, $\mathbf{v}_i'$, which is conflict-free with the other $n-1$ vehicles if vehicle i is separated from the other vehicles such that $\sigma_i \leq \rho_i$.

Proof In order for a conflict-free point to exist, the collision cones from the other vehicles cannot completely cover the admissible disk of possible velocities of vehicle i. The worst case for this coverage when the other vehicles are capable of higher speed than vehicle i is when all the vehicles have the same heading at their maximum speed, and their positions splay out their collision cones so as to form one large, continuous cone (see Fig. 7). The interior angle of this large cone must be less than $2\arcsin(v_{i,max}/v_{max})$ to ensure that it cannot cover the entire admissible circle. When vehicle i's speed is higher than the other vehicles, the arcsine

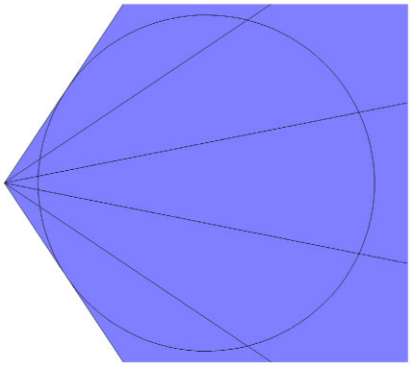

Fig. 7 Example of a borderline case for six vehicles where no conflict-free maneuver exists. As in Fig. 6, the *circle* is the edge of the allowable disk (radius $v_{i,max}$), and the *shaded regions* (*blue* in the online color version) are the collision cones, shown with all equal velocities, of magnitude greater than $v_{i,max}$

takes on its limiting value of $\pi/2$ and becomes conservative because it only allows the combined collision cone to cover a half-space within the disk. Together these constraints form (26). □

In the special case where all of the vehicles are exactly the same distance from vehicle i (as in Fig. 7), the bound (26) can be simplified to a bound on distance. First note that the sum of the collision cones is just $(n-1)\alpha_e$, and rearranging (22) gives $\|\tilde{\mathbf{r}}\| \geq (d_{sep}+\delta)/\sin\alpha_e$. Then the distance bound is

$$\|\tilde{\mathbf{r}}\| \geq \begin{cases} \dfrac{d_{sep}+\delta}{\sin(\frac{\arcsin(\frac{v_{i,max}}{v_{max}})}{n-1})}, & v_{i,max} < v_{max}, \\ \dfrac{d_{sep}+\delta}{\sin(\frac{\pi}{2(n-1)})}, & v_{i,max} \geq v_{max}. \end{cases} \tag{27}$$

Note that as n gets large, the bound approaches a linear relationship with $n-1$. Intuitively, as more vehicles exist in a space, they must be spaced out more to ensure a safe trajectory exists between them.

3.3 Algorithm

The first step of the DRCA algorithm is to decide when a vehicle needs to perform its deconfliction maneuver. Let $\mathscr{D}$ be the set of vehicles which are currently deconflicting (either performing the deconfliction maneuver or deconfliction maintenance). Each vehicle in $\mathscr{D}$ must be able to broadcast to all the other vehicles in $\mathscr{D}$.

Vehicle i follows its desired control until $\rho_i - \sigma_i < \phi$, as defined in (25) and (26), where ϕ is a small positive number chosen to be small enough to give the vehicle time to compute its optimization before $\sigma_i > \rho_i$. At this point, vehicle i becomes a member of $\mathscr{D}$ and performs its deconfliction maneuver, which amounts to choosing a conflict-free velocity vector, broadcasting it to the group, and then accelerating to attain it as quickly as possible. Once this vehicle attains a conflict-free state with the other vehicles in $\mathscr{D}$, it switches to deconfliction maintenance. The other vehicles in $\mathscr{D}$ that are already performing deconfliction maintenance avoid this new vehicle using its broadcast velocity instead of its actual velocity until its maneuver is complete. Conceptually, this method is akin to a turn signal, where the new vehicle tells the group how it will be accelerating, and the group can then work around that decision. As such, each vehicle will see no conflicts with a newly added member of the group.

If $\mathscr{D}$ is empty, then each vehicle uses $\sigma_i = \alpha_e$ of the nearest vehicle. The first vehicle to reach $\rho_i - \sigma_i < \phi$ makes itself the first member of $\mathscr{D}$ but does not perform a deconfliction maneuver, since the other vehicle will be able to safely deconflict even when the vehicles are closer together. Now that $\mathscr{D}$ is nonempty, the system then follows the previous directions. Once $\rho_i - \sigma_i > \phi$ again, vehicle i can cease to be in $\mathscr{D}$ and start solely following its desired control again.

It is possible for multiple vehicles to cross the threshold for starting their maneuvers at the same time. However, their maneuvers must be consecutively arranged. A simple, distributed method is for each vehicle to broadcast a random number, then the one with the lowest number computes its new velocity first and broadcasts it, allowing the next vehicle to compute its maneuver using that vehicle's intended velocity, and so on.

The only information the DRCA algorithm needs on a continuous basis is the position and velocity of each vehicle in $\mathscr{D}$, which can either come from broadcast communication (e.g. a transponder) or sensing (e.g. radar). If the system is heterogeneous, then the vehicles must know a bound on the maximum speed and the minimum separation distance of the other vehicles in $\mathscr{D}$, which can be accomplished by a broadcast communication or vehicle identification. Finally, the vehicles must know who is in $\mathscr{D}$ and must receive the intended velocity of any vehicle performing a deconfliction maneuver. This information cannot be attained through sensing, so some form of broadcast communication is necessary to disseminate it. Any vehicle not yet in $\mathscr{D}$ only needs the information at a range just outside of where $\rho_i - \sigma_i = \phi$. In the case of $\mathscr{D}$ being empty, this information is required only of the nearest vehicle.

Two caveats exist with respect to the assumptions in this algorithm. First, two separate deconfliction groups cannot safely merge. The reason is that vehicles performing deconfliction maintenance are in general separated by as little as d_{sep}, for which $\sigma_i > \rho_i$, but vehicles joining $\mathscr{D}$ must have $\sigma_i < \rho_i$. In many applications, vehicles are added to a group singly. For example in an airport, if one considers the congested airspace of the airport itself as a group, wherein all vehicles are in the deconfliction maintenance phase with each other, then new aircraft arrive from all around where

the airspace is not as congested. These new aircraft can safely integrate one at a time as long as they stay separated from each other by enough distance that when one vehicle joins $\mathscr{D}$, it does not cause $\sigma_i > \rho_i$ for other vehicles outside the group.

The second caveat concerns communication range. While the communication range must be chosen such that vehicle i can talk to vehicle j by the time $\rho_i - \sigma_i = \phi$, this range only ensures that $\mathscr{D}$ is connected, not necessarily all-to-all. This problem can be overcome by having vehicles rebroadcast information they have received. This multi-hop method will tend to accrue more time delay, however the robustness analysis in Sect. 4 shows how the algorithm can be made robust to these delays.

3.4 Heuristic extensions

The guarantees provided by the maneuver are ideal if one can ensure the vehicles always have sufficient warning of each other's presence, but this may not be the case for all systems. However, if at any time the system can come to a conflict-free state, the strong guarantee of Theorem 1 will still apply. Therefore the robustness of the system to violations of the conditions of Theorems 2 and 3 can be increased by choosing heuristic behaviors for the deconfliction maneuver in cases where it would otherwise be undefined.

The first thing that could cause a solution to not exist is if $\|\tilde{\mathbf{r}}\| < d_{sep} + \delta$, causing α_e to be undefined. In this situation, let the heuristic be to set $\alpha_e = \pi/2$ (its limiting value). The condition (26) is conservative, so even if it is not met, the optimization should be attempted because a solution is still likely to exist. In the event that no solution exists, let the heuristic return the solution $\mathbf{v}_i' = -\mathbf{v}_i$. This solution was chosen because it is a velocity the vehicle can attain, and it will take significant acceleration to get there. The intuition behind the choice is that when vehicles are in conflict, it is better to do something than nothing, and by continuing to use maximum acceleration, there is more chance that the system will stumble upon a conflict-free state.

If a vehicle does not find a solution to its optimization, it does not join $\mathscr{D}$ unless it attains a conflict-free state, since the deconfliction maintenance controller that the other vehicles in $\mathscr{D}$ are running does not allow for vehicles in conflict. As a worst case, it is better for the new vehicle to collide with one of the members of $\mathscr{D}$ than to cause all of the members of $\mathscr{D}$ to collide with each other.

4 Analysis

Liveness and robustness properties of the DRCA algorithm will be proven in the following subsections. Robustness for this system is split into two categories: robust stability and robust avoidance. Robust stability encompasses the traditional ideas of linear stability analysis, focused on how much gain can be applied to the system before time delays or other unmodeled dynamics drive the system to oscillate explosively. Robust avoidance is a higher-level idea relating to how relaxation of the assumptions (in this case cooperation) affects the validity of the collision avoidance guarantees.

4.1 Liveness

Proving that the DRCA algorithm allows all vehicles to attain their goals in general is made difficult by the breadth of possible goals. For instance, if the desired control is for two vehicles to attain the same waypoint, the DRCA algorithm will always choose safety over performance, so the goal will not be met in favor of keeping the vehicles from colliding. Likewise, if the desired controller is a cooperative search algorithm, it becomes unclear how one would define the liveness of a solution. In light of these facts, liveness will be shown here for a strict set of assumptions, though the arguments should hold qualitatively for a wider range of situations.

Theorem 4 *Let the deconfliction group, $\mathscr{D}$, contain n constant-speed vehicles that are not in conflict. Let their desired controllers be heading regulators where each vehicle has some constant desired heading. If each vehicle has an avoidance gain, k_n, such that its speed is separated from the others' speeds by at least the avoidance threshold, ε_n, then all of the vehicles will attain their desired headings while remaining conflict-free.*

Proof Theorem 1 already guarantees that the system will remain conflict-free for all time. Any vehicle for which $p_n \geq \varepsilon_n$ will follow its desired control exactly and, hence, will attain its heading as long as the collision cones remain at least ε away.

Any vehicle for which $p_n < \varepsilon_n$ must have $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$, because if $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} \leq 0$ then from (8), $\mathbf{e} = \tilde{\mathbf{v}}$ and $\|\tilde{\mathbf{v}}\| \geq \varepsilon_n$ and since $p_n \geq \|\mathbf{e}\|$, then $p_n \geq \varepsilon_n$, which is a contradiction. Since $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$, (18) applies. A collision cone only acts as a barrier if the desired controllers want to accelerate the vehicles such that the $\hat{\mathbf{e}}^\mathsf{T}\dot{\tilde{\mathbf{v}}}$ term is negative, since the DRCA algorithm adjusts those inputs such that $\dot{m} \geq 0$. However, we already know that the second term of (18) is strictly positive since the vehicles are not in conflict, and $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$ (see proof of Theorem 1). Therefore $\hat{\mathbf{e}}^\mathsf{T}\dot{\tilde{\mathbf{v}}}$ can be negative while allowing the DRCA algorithm to keep $\dot{m} \geq 0$. Recall from (15) that $\hat{\mathbf{e}}_{ij}^\mathsf{T}\dot{\tilde{\mathbf{v}}}_{ij} = u_{n_i}\hat{\mathbf{e}}_{ij}^\mathsf{T}\hat{\mathbf{n}}_i + u_{n_j}\hat{\mathbf{e}}_{ij}^\mathsf{T}\hat{\mathbf{n}}_j$. Even though each vehicle wants its own term to be negative, and even though the sum is allowed to be negative, the DRCA algorithm may cause one vehicle's term to be positive (move away from its goal) in order to allow the other to be more negative (approach its

goal faster). However, once one vehicle attains its goal heading, then its u_n goes to zero, which allows the other vehicle to start moving toward its goal.

Even though these conflicts can be daisy-chained such that each vehicle is in conflict with another vehicle, this analysis still guarantees that at least one vehicle in the group is moving toward its goal at all times, and so some vehicle must eventually attain its goal and allow another to progress. Therefore all the vehicles in the group must eventually attain their desired headings. □

This theorem may seem limited, but attaining a fixed heading is effectively equivalent to moving towards a distant waypoint, which is a common goal for a wide range of applications. Additionally, even for changing desired headings, this analysis still shows some net progress will be made, but how that progress will be distributed between the vehicles is unclear.

The main limiting assumption in this theorem is the speed separation. The reason for this assumption is to guarantee that $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0$, because otherwise $\dot{m} = \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}}$, and there may be no net progress. This situation can only occur if the relative velocity is small: $\|\tilde{\mathbf{v}}\| < \varepsilon$. Most desired controllers tested so far by the authors on equal-speed vehicles or variable-speed vehicles have shown this situation to be unstable, such that the vehicles will move to a different configuration where progress toward goals can be made. However, it is possible to stabilize this situation and form a livelock, which must be avoided in the design of a desired controller or else goals may never be attained.

One whole class of desired controllers is eliminated by this logic: flocking or schooling controllers. The goal of these controllers is to make $\tilde{\mathbf{v}} = 0$, which amounts to $m = 0$, which the DRCA algorithm does not allow. The vehicles cannot progress toward their goal of $\tilde{\mathbf{v}} = 0$ because the DRCA algorithm sees this situation as being on the edge of safety. On the plus side, for vehicles which can stop, a deadlock is impossible because only one vehicle is allowed to stop at a time (since two stopped vehicles would also have $m = 0$). However, avoiding deadlock is not sufficient to prove liveness in general.

4.2 Robust stability

As complex as this control algorithm may appear, it can still be analyzed for robustness using linear tools. The deconfliction maneuver creates a constant input, so robust stability does not apply. The desired controller is assumed to exhibit an adequate degree of robust stability, since this controller is designed for the system in question. Therefore, the deconfliction maintenance controller is the part of this algorithm that needs to be analyzed for robust stability. Depending on the system in question, uncertainties can take many different forms. The following analysis is based upon a complex multiplicative uncertainty bounded by the transfer function $w(j\omega)$.

Stability in this case is analyzed about an equilibrium in the conflict space, i.e. where $\dot{m} = 0$ for the pair of vehicles in question. This equilibrium refers to the period from when the vehicles end their deconfliction maneuver to when they begin to pass each other, characterized by nearly constant-velocity trajectories.

Theorem 5 *Let $k_{ij} = k_{t_i} + k_{n_i} + k_{b_i} + k_{t_j} + k_{n_j} + k_{b_j}$, where k is the gain from (12). The system is stable in the presence of a complex multiplicative uncertainty bounded by $w(j\omega)$ if k_{ij} satisfies*

$$\frac{k_{ij}}{\sqrt{k_{ij}^2 + \omega^2}} < \frac{1}{|w(j\omega)|} \quad \forall\omega. \tag{28}$$

Proof Recalling (14), (15), and (18), one has that

$$\dot{m}_{ij} = m_{ij}\left(\frac{u_{t_i}}{p_{t,ij}} + \frac{u_{n_i}}{p_{n,ij}} + \frac{u_{b_i}}{p_{b,ij}} + \frac{u_{t_j}}{p_{t,ji}} + \frac{u_{n_j}}{p_{n,ji}} + \frac{u_{b_j}}{p_{b,ji}} + h_{ij}\right), \tag{29}$$

where h_{ij} is a positive quantity given by

$$h_{ij} = \begin{cases} \frac{\|\tilde{\mathbf{v}}\|\cos(|\beta|-\alpha)}{\|\tilde{\mathbf{r}}\|\cos\alpha}, & |\beta| - \alpha \le \frac{\pi}{2}, \\ 0, & |\beta| - \alpha > \frac{\pi}{2}. \end{cases}$$

The worst case for robust stability is when the feedback gain is the largest (most negative), and one can see by looking at Fig. 5 that the largest gain in terms of p^+ will occur when $p^- = \varepsilon$ and $u_d = u_{min}$. One can assume without loss of generality that the nearest conflict is in the positive direction, i.e. $p > 0$. In this case,

$$\begin{aligned} m\frac{u_t}{p_t} &= m\frac{F(p_t,\varepsilon)}{p_t} \\ &= m\left(\frac{u_{t,max}}{p_t} + \frac{u_{t,min} - u_{t,max}}{\varepsilon_t}\right) \\ &= -k_t m + \hat{\mathbf{e}}^{\mathsf{T}}\hat{\mathbf{t}}u_{t,max}. \end{aligned}$$

Applying this result to (29) yields

$$\dot{m}_{ij} = -\left(k_{ij} - h_{ij}\right)m_{ij} + \hat{\mathbf{e}}^{\mathsf{T}}\Theta_i\mathbf{u}_{i,max} + \hat{\mathbf{e}}^{\mathsf{T}}\Theta_j\mathbf{u}_{j,max}, \tag{30}$$

where $\mathbf{u}_{i,max} = [u_{t_{i,max}}, u_{n_{i,max}}, u_{b_{i,max}}]^{\mathsf{T}}$. The second two terms are always positive ($p_t > 0 \implies \hat{\mathbf{e}}^{\mathsf{T}}\hat{\mathbf{t}} > 0$, etc.) and only serve to push the equilibrium to a value of m that is greater than zero. Only the first term has bearing on the robust stability of the system. One can open the loop on this

negative feedback system and examine the loop gain as a transfer function, $L(s)$:

$$L(s) = \frac{k_{ij} - h_{ij}}{s}.$$

Note that $h_{ij} < k_{ij}$, or else this system does not have an equilibrium, and the stability analysis does not apply.

The complementary sensitivity function, $T(s)$, is given by

$$T(s) = \frac{L}{1+L} = \frac{k_{ij} - h_{ij}}{s + k_{ij} - h_{ij}}. \tag{31}$$

For a system perturbed by a complex multiplicative uncertainty, the perturbed loop gain, $L_p(s)$, is given by

$$L_p(s) = L(1 + w\Delta), \qquad |\Delta(j\omega)| \leq 1, \quad \forall\omega.$$

In this case, robust stability to an uncertainty bounded by w is guaranteed if

$$|T(j\omega)| < \frac{1}{|w(j\omega)|} \quad \forall\omega,$$

as shown by Skogestad and Postlethwaite (2005). From (31),

$$|T(j\omega)| \leq \frac{k_{ij} - h_{ij}}{\sqrt{(k_{ij} - h_{ij})^2 + \omega^2}}.$$

Since $0 \leq h_{ij} \leq k_{ij}$, then

$$\frac{k_{ij} - h_{ij}}{\sqrt{(k_{ij} - h_{ij})^2 + \omega^2}} \leq \frac{k_{ij}}{\sqrt{k_{ij}^2 + \omega^2}},$$

which implies that (28) is a sufficient bound for stability. □

Depending on the nature of the system on which this algorithm is implemented, the uncertainty, w, could take many forms. As an example, one common type of unmodeled dynamics is time delay. Delays can be caused by sensors, communication, or even the discretization of this continuous-time system.

Lemma 3 *The system is stable in the presence of a constant, pure time delay no greater than τ if*

$$k_{ij} < \frac{1.47}{\tau}. \tag{32}$$

Proof A multiplicative uncertainty, w, that bounds this time delay is given by Skogestad and Postlethwaite (2005):

$$w(\omega) = \begin{cases} \left|1 - e^{-j\omega\tau}\right|, & \omega < \pi/\tau, \\ 2, & \omega \geq \pi/\tau. \end{cases}$$

When $\omega \geq \pi/\tau$, then (28) becomes

$$\frac{k_{ij}}{\sqrt{k_{ij}^2 + \omega^2}} < \frac{1}{2},$$

$$k_{ij} < \frac{\omega}{\sqrt{3}} < \frac{\pi}{\tau\sqrt{3}}.$$

If $\omega < \pi/\tau$, then

$$\frac{k_{ij}}{\sqrt{k_{ij}^2 + \omega^2}} < \frac{1}{|1 - e^{-j\omega\tau}|},$$

$$\frac{k_{ij}}{\sqrt{k_{ij}^2 + \omega^2}} < \frac{1}{\sqrt{2 - 2\cos(\omega\tau)}},$$

$$k_{ij}^2 (1 - 2\cos(\omega\tau)) < \omega^2,$$

$$k_{ij}\tau < \frac{\omega\tau}{\sqrt{1 - 2\cos(\omega\tau)}}.$$

The right hand side is now in terms of a single variable ($\omega\tau$), and the minimum can be calculated numerically:

$$\min_{\omega\tau} \frac{\omega\tau}{\sqrt{1 - 2\cos(\omega\tau)}} \approx 1.4775.$$

This case is more restrictive because $1.4775 < \pi/\sqrt{3}$. Therefore (32) is sufficient to guarantee stability in the presence of the time delay τ. □

This type of analysis can be used to find similar bounds on the gain of the system to guarantee stability to other types of unmodeled dynamics and uncertainty, for instance noise or lags caused by filtering.

4.3 Robust avoidance

A possible challenge for this system is a vehicle which does not cooperate, i.e. does not satisfy (20). Such a vehicle is regarded as antagonistic, regardless of its actual goals. Note that a vehicle which does not run the DRCA algorithm, but rather holds a constant velocity always satisfies (20) and is therefore not considered antagonistic.

One can see that in a multivehicle scenario, several antagonistic vehicles could surround another vehicle's velocity vector with their collision cones, bringing them together until no conflict-free region was left, leaving the hapless vehicle with no guarantee of collision avoidance. Indeed, in some situations collision avoidance may be impossible against adversaries. Likewise even a single adversary can pin another vehicle between itself and one or more constant-velocity vehicles (or vehicles with very small control authority). This problem is fundamental to pursuit-evasion within groups of vehicles, and this algorithm does not provide an

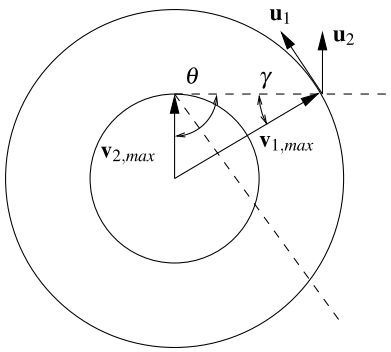

Fig. 8 Geometry of the worst situation for avoiding an adversary. The *dotted lines* represent the collision cone, and the *circles* represent the set of allowable velocity vectors for the two vehicles

easy solution. However, for a two-vehicle system with a single adversary, the DRCA algorithm still provides a guarantee.

Theorem 6 *Consider a two-vehicle system in which the vehicle's dynamics are restricted by* (3). *Assume vehicle one runs the deconfliction maintenance controller, and vehicle two is an adversary using an unknown controller. The vehicles will remain conflict-free for all time provided they started that way and that*

$$u_{1,max} \geq \frac{u_{2,max} v_{1,max} \sqrt{3}}{\sqrt{v_{1,max}^2 - v_{2,max}^2}}, \tag{33}$$

which implies also that $v_{1,max} > v_{2,max}$.

Proof The control function (10) ensures that at the edge of a conflict, not only does the deconflicting vehicle not approach the conflict any further, but it actually applies its maximum control authority in the opposite direction. Rederiving (30) for only vehicle one using deconfliction maintenance yields

$$\dot{m} = -(k_t + k_n + k_b - h)\, m + \hat{\mathbf{e}}^{\mathsf{T}} \Theta_i \mathbf{u}_{1,max} + \hat{\mathbf{e}}^{\mathsf{T}} \Theta_j \mathbf{u}_2,$$

where one can see that the first term goes to zero as $m \to 0$. The second term is always greater than or equal to $u_{1,max}$ because $\hat{\mathbf{e}}^{\mathsf{T}} \hat{\mathbf{t}}_1 \geq 0$, etc. However, the last term is upper-bounded by $u_{2,max}\sqrt{3}$, so $u_{1,max} \geq u_{2,max}\sqrt{3}$ is sufficient to guarantee $\lim_{m \to 0^+} \dot{m} \geq 0$, and thus that conflict is avoided.

A problem with this analysis is that the control authority limits of the vehicles are not actually fixed, but become restricted when the maximum speed, v_{max}, is reached. The worst case occurs when vehicle one is at its maximum speed and is on the edge of a conflict with vehicle two, which is nearly at its own maximum speed. The angle that the collision cone makes with the boundary of vehicle one's allowable velocity set defines the ratio of control authority needed to avoid vehicle two's worst action. As can be seen in Fig. 8, the worst angle for the collision cone to have is when γ is maximized. The law of sines gives

$$\frac{\sin\gamma}{v_{2,max}} = \frac{\sin\theta}{v_{1,max}},$$

so γ is maximized when $\sin\theta = 1$, and the maximum is $\gamma = \arcsin(v_{1,max}/v_{2,max})$. The required control authority to both stay out of conflict and to stay within $v_{1,max}$ is then

$$u_{1,max} \geq u_{2,max} \sec\gamma \sqrt{3}.$$

Substituting the maximum γ from above and simplifying yields the bound (33). □

5 Performance

In this section the performance of the DRCA algorithm will be demonstrated in simulation. The following subsections are intended to illustrate a variety of possible applications as well as showcase the nominal and off-nominal behavior of the algorithm. The first subsection shows the standard way the algorithm functions, conforming to all of the requirements for the guarantees of safety in a 2D, variable-speed, heterogeneous system. The next subsection demonstrates how vehicle priority can be changed and how it affects the trajectories. Section 5.3 shows a number of simulations where various requirements of the guarantees are broken and demonstrates the limits of the heuristic performance of the algorithm. Section 5.4 discusses how the DRCA algorithm can be adjusted to work with swarms of vehicles and shows an example. The final subsection compares the DRCA algorithm to a Satisficing Game Theory (SGT) algorithm in two classically difficult situations. Only this last subsection uses real-world scale; the other simulations use scales near unity because their purpose is to show the generic behaviors of the algorithm, rather than a particular application.

5.1 Variable speed

This simulation (Fig. 9) is of a heterogeneous group of five vehicles performing target tracking. Each vehicle has a target that moves at constant velocity (0.4 m/s), whose position the vehicle is attempting to attain using a standard target following controller. Circles around each vehicle in Fig. 9a denote their size, and overlapping circles imply collision. In this example $k = 5\ \mathrm{s}^{-1}$, each vehicle's maximum speed is 0.5 m/s, and its maximum control authority is $0.2\ \mathrm{m/s}^2$. Figure 9e shows the excess separation distances for each pair of vehicles, demonstrating that no collisions occur.

The initial conditions in this example are contrived such that the vehicles enter the deconfliction group one at a time,

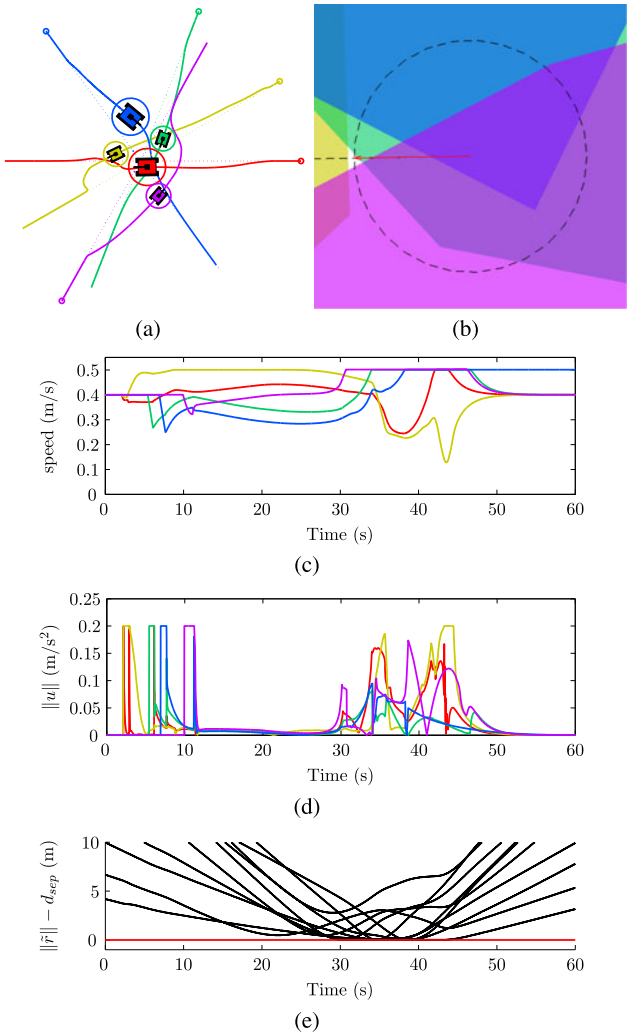

Fig. 9 Five vehicle simulation: (**a**) shows the vehicle trajectories. Target paths are shown as dotted lines. Vehicle images are plotted at 30 seconds, while their initial positions are denoted by small circles. (**b**) shows a snapshot of the collision cones from the perspective of the *center* vehicle at 30 seconds. The *center arrow* denotes the *center* vehicle's velocity. This illustration effectively overlaps several diagrams like Fig. 4. The vehicle speeds and control magnitudes are shown in (**c**) and (**d**), respectively. The excess separation of the vehicles is plotted in (**e**)

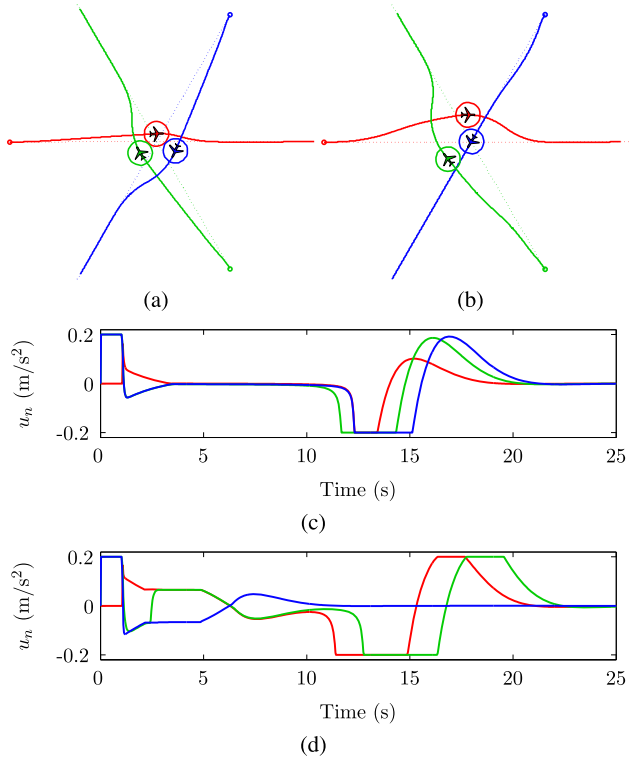

Fig. 10 Preferential routing comparison for three vehicles. Desired paths are depicted by *dotted lines*. The symmetric case is shown in (**a**) where each vehicle has $k_n = 5\ \text{s}^{-1}$, while (**b**) shows the difference when the *right-hand* vehicle (*blue* in the online color version) is given routing preference ($k_n = 10\ \text{s}^{-1}$)

counter-clockwise starting from the center vehicle (red in the online color version). Each vehicle starts its variable-speed deconfliction maneuver just before the bound (26), to ensure all the safety guarantees are met. The deconfliction maneuvers can be seen as short plateaus in the control magnitude plot (Fig. 9d).

The snapshot of the simulation in Figs. 9a and b shows just how small a conflict-free region can be. While Theorem 1 guarantees that this region will never disappear, one can see how if any of the smaller vehicles (green, yellow or purple in the online color version) became antagonistic, they could easily cause a collision despite the best efforts of the other vehicles to deconflict. This type of situation is why the analysis of antagonistic vehicles was limited to only two-vehicle scenarios.

5.2 Preferential routing

A feature of the DRCA algorithm is that vehicles can be given different priority by adjusting the gain, k (12), of one vehicle relative to the other vehicles in the group. Vehicles with lower gain values will effectively get out of the way of higher gain vehicles, allowing those with priority to follow their desired controllers more closely.

In order to isolate the effects of preferential routing, the pair of simulations in Fig. 10 use identical vehicles and initial conditions. In this case the three vehicles are restricted to a constant speed of 1 m/s. The vehicles start in an exact conflict (i.e. without control, all would reach the same point at the same time), and immediately begin their constant-speed deconfliction maneuver. The upper vehicle traveling left to right (red in the online color version) is the first vehicle, and so maintains its heading throughout the maneuver. After one second, a conflict-free state is attained, after which the vehicles attempt to return to their original straight paths.

Even though this simulation is symmetric and homogeneous, the upper vehicle stays closer to its path than the

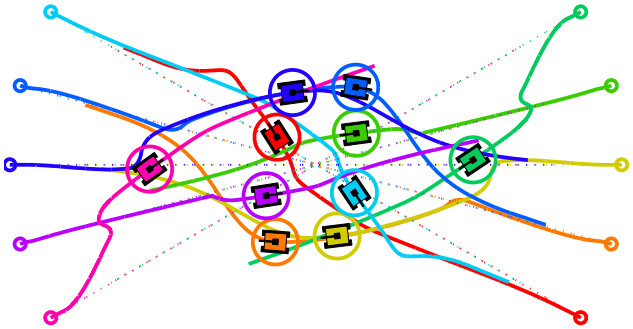

Fig. 11 Merging two groups of vehicles. No collisions occur despite the vehicles being too close for all the safety guarantees to apply

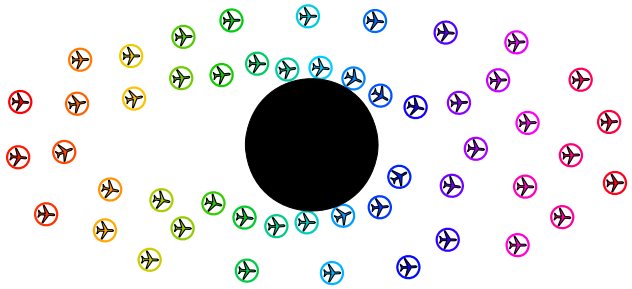

Fig. 12 Fifty-vehicle swarm avoiding a static obstacle

others because it had order preference during the deconfliction maneuver. In Fig. 10b, the right-hand vehicle (blue in the online color version) was given priority by doubling its gain, k_n, relative to the others. As can be seen, the right-hand vehicle still performs the same deconfliction maneuver, but once a conflict-free state is achieved, it begins to turn back toward its path. The other vehicles are forced to make way for it, while never allowing a conflict to form.

5.3 Heuristic performance

Unlike the previous examples, this simulation (Fig. 11) violates some of the guarantees of safety, forcing the algorithm to use its heuristic backup to try and avoid collisions. Ten vehicles are arranged in two opposing groups. The minimum separation is $d_{sep} = 2$ m, their communication range is 8.8 m, their maximum speed is 1 m/s, their target speed is 0.6 m/s, and their maximum control authority is 0.5 m/s^2, giving these variable-speed vehicles $\delta = 8$ m (23).

The first problem is that the nearest vehicles have $\|\tilde{\mathbf{r}}\| < d_{sep} + \delta$, so α_e is set to $\pi/2$, rather than forcing it to be undefined. The second problem is that the vehicle communication range is only 10% more than $d_{sep} + \delta$. These vehicles are set up to rebroadcast information about their neighbors, such that any connected group will have full information (pseudo all-to-all), which means that the two groups know about their own members but not about each other until they get close enough to communicate. This leads to two deconfliction maneuvers: one at the beginning to avoid the other vehicles in the group, and later a second to avoid everyone. Despite these difficulties, the algorithm successfully deconflicts the vehicles such that no collisions occur.

5.4 Formations

One limitation of the DRCA algorithm is that vehicles moving with identical velocities are on the edge of conflict ($m = 0$). Because the deconfliction maintenance controller is continuous, it will tend to push the vehicles to an equilibrium where $m > 0$, which means the vehicles move slightly away from one another. This behavior precludes the use of the DRCA algorithm for collision avoidance within a formation of vehicles because the desired state is for all of the vehicles to move with the same velocity.

One way around this problem is to split up the duties of the desired controller and the DRCA algorithm. In the case of formation flight, the relative motion of the vehicles can be stabilized with any of a variety of controllers from the literature (Reif and Wang 1999; Skjetne et al. 2002; Gazi 2006). By using a desired controller that guarantees intervehicle spacing, the DRCA algorithm can simply be turned off between the vehicles in the formation. However, for the more difficult task of avoiding objects moving at high relative speed, the DRCA algorithm can take over.

A simulation of this situation for a fifty-vehicle swarm avoiding a static obstacle is shown in Fig. 12. In this case, the communication topology is broken into two pieces: the DRCA algorithm uses a star graph, from the static obstacle to each other vehicle, while the desired controller uses everything else (the complete graph minus the star graph). Therefore the desired controller has no knowledge of the static obstacle; it simply keeps the vehicles spaced out using potential functions. In this way the vehicles avoid the static obstacle in a guaranteed fashion using the DRCA algorithm while the desired controller maintains intervehicle spacing.

The maximum vehicle speed is 1 m/s while the swarm moves at 0.5 m/s, and the vehicle maximum control authority is 0.2 m/s^2. The vehicle diameters are 1 m, and the obstacle's diameter is 6 m. The vehicles begin their deconfliction maneuver when they have 6.5 m of excess separation from the obstacle (13 m center-to-center). The swarm is "unzipped" around the obstacle, but the vehicles avoid it by no more than they have to. One can also choose the distance at which vehicles deconflict such that α_e is no more than some threshold, thus keeping the $\|\Delta\mathbf{v}\|$ of the maneuver small enough to ensure the desired controller can adequately maintain intervehicle spacing.

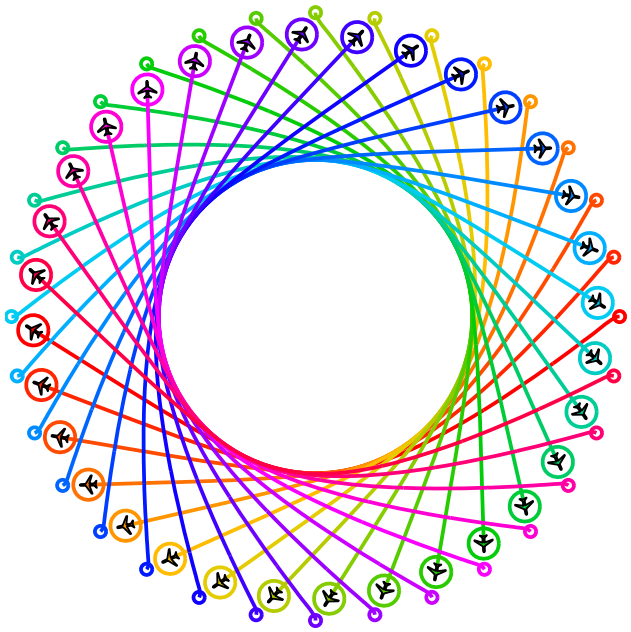

Fig. 13 Collision avoidance of 32 aircraft using the DRCA algorithm. Vehicles are shown 30 seconds before attaining their waypoints. No near misses occur even though the vehicles start without sufficient separation to guarantee deconfliction (26)

5.5 Comparison to an SGT algorithm

To demonstrate the performance strength of the DRCA algorithm, two systems were simulated to compare with the results of a Satisficing Game Theory algorithm by Hill et al. (2005). The first scenario is based originally on a model used by Pallottino et al. (2002), aimed at air traffic control. The aircraft have a constant speed of 500 mph, start evenly spaced on a circle of radius 50 miles, pointed at the center, and each is attempting to reach a waypoint on the exact opposite side of the circle (the starting point of the opposite aircraft). This scenario is referred to as a choke point because, without deconfliction, all the vehicles would meet at the center (Fig. 13).

A near miss is defined as two aircraft coming within five miles of each other, so d_{sep} was set to this value. The efficiency of the maneuver is defined as the average of the percentage of time delay in arrival at the waypoints:

$$\text{Efficiency} = \frac{1}{n}\sum_{i=1}^{n}\frac{T_r}{T_i},$$

where $T_r = 12$ minutes is the reference time (flying straight), and T_i is the actual time taken for the ith vehicle. In Hill et al. (2005), the maximum turn allowed was 5 degrees per second, which corresponds to $u_{max} = 64$ ft/s^2. Additionally, the gain chosen was $k_n = 0.4$ s^{-1}.

In order to fairly compare these two algorithms, the DRCA algorithm has been set up for this simulation such that each vehicle has information only from those vehicles within a 50 mile radius of itself, just as in Hill et al. (2005). While this breaks some of the guarantees of safety, the heuristic performance will be shown to be more than adequate. Additionally, since the vehicles start off such that $\|\tilde{\mathbf{r}}\| < d_{sep} + \delta$, this simulation used half of the standard δ value, because the safety guarantees are already gone, so that the choice of δ such that $\|\tilde{\mathbf{r}}\| > d_{sep} + \delta$ is appropriate.

Figure 13 shows a simulation of the DRCA algorithm deconflicting 32 vehicles, the densest choke pattern demonstrated by Hill et al. (2005). The DRCA algorithm attained an efficiency of 87.6%, compared to 85.7% in Hill et al. (2005). The real achievement was that no near misses occurred with the DRCA algorithm, as opposed to 19 in Hill et al. (2005). Even though each vehicle only knows about other vehicles within 50 miles, the symmetry of this situation means that the closest conflicts for each vehicle are only with its two nearest neighbors, and so the communication limitations are not noticeable in this simulation.

The second scenario is that of two perpendicular flows of the same aircraft as before. Like the worst case shown in Hill et al. (2005), the aircraft streams are constant speed, with a starting separation of seven miles between the aircraft. Because the DRCA algorithm does not allow vehicles to maintain the same velocity as each other, the algorithm was modified such that vehicles only avoid each other within a circle of radius 50 miles from the initial crossing point. That is, for two vehicles to detect each other, they must both be in the circle and be within 50 miles of each other. Unlike the standard implementation of the DRCA algorithm, these vehicles must frequently perform deconfliction maneuvers as new vehicles are spotted. The δ used is 1 mile, but everything else is the same as the previous simulation.

The goal of each vehicle in the simulation in Hill et al. (2005) was unclear, so for the simulation in Fig. 14 the desired controller points each vehicle toward a distant waypoint along their original trajectory, which is effectively equivalent to attaining their original heading. Likewise, the definition of efficiency was unclear in this example, and so is omitted. The DRCA algorithm demonstrated no near misses, while Hill et al. (2005) recorded 29. The algorithms were qualitatively similar, forming alternating waves of vehicles that flowed past each other.

6 Conclusion

The primary contribution of this paper is a widely applicable, provably safe collision avoidance algorithm. The DRCA algorithm is designed for the exceedingly simple 3D double-integrator, which can model a vast array of vehicles through the application of appropriate control saturation functions. Instead of using more complex dynamics to model nonholonomic constraints, state-dependent control saturation can be

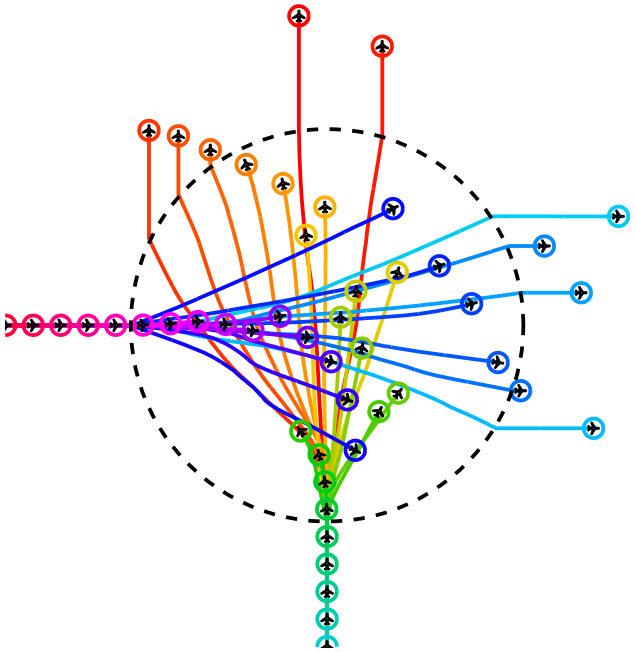

Fig. 14 Collision avoidance of two perpendicular flows. Vehicles only perform the DRCA algorithm when they are inside of the black dashed circle (radius 50 miles)

used to accomplish the same goal. As shown in the robustness analysis, higher order dynamics can be accounted for through the proper choice of gains.

The most important aspects of the DRCA algorithm are the complete guarantees of safety at every phase. Almost no other collision avoidance algorithms in the literature give complete safety guarantees for n vehicles in the presence of acceleration or force limitations. Even the other optimization schemes frequently give no mention of when a feasible solution exists. Strict requirements are used by the guarantees in this paper, but those requirements help to shed light on the fundamental constraints of avoiding collisions with limited available acceleration.

As the name Distributed Reactive Collision Avoidance implies, this algorithm ensures real-time operability by distributing the computations among the agents, bounding the computational scaling, and requiring no central server. The information required by this algorithm in order to maintain safety guarantees is higher than for some others and may be higher than is practical for certain systems. However, it has been demonstrated that even when these requirements are relaxed, for instance by making the system truly decentralized with only local information, the heuristic performance is still better than other algorithms. Part of the reason is that although the deconfliction maneuver may not be proven safe, if the system becomes conflict-free, the deconfliction maintenance controller will guarantee it stays that way.

A lengthier description of this material, including a constant-speed deconfliction maneuver, is available in the authors' dissertation, Lalish (2009). Recently the DRCA algorithm was successfully implemented on quadrotor helicopters (Melander 2010) and robotic fish (Melander et al. 2010), demonstrating its practicality.

Acknowledgements This work supported in part by National Science Foundation grant CMS-0234861 and in part by AFOSR grant FA-9550-07-1-0528.

References

Carbone, C., Ciniglio, U., Corraro, F., & Luongo, L. (2006). A novel 3D geometric algorithm for aircraft autonomous collision avoidance. In *Proc. IEEE conference on decision and control*.

Chakravarthy, A., & Ghose, D. (1998). Obstacle avoidance in a dynamic environment: A collision cone approach. *IEEE Transactions on Systems, Man, and Cybernetics*, *28*(5), 562–574.

Eby, M., & Kelly, W. (1999). Free flight separation assurance using distributed algorithms. In *Proc. IEEE aerospace conference* (pp. 429–441).

Eurocontrol (2007). *Eurocontrol specifications for the use of military unmanned aerial vehicles as operational air traffic outside segregated airspace* (Version 1.0, Doc No. EUROCONTROL-SPEC-0102).

Fasano, G., Accardo, D., & Moccia, A. (2008). Multi-sensor-based fully autonomous non-cooperative collision avoidance system for unmanned air vehicles. *Journal of Aerospace Computing, Information, and Communication*, *5*, 338–360.

Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, *17*(7), 760–772.

Frazzoli, E., Mao, Z., Oh, J., & Feron, E. (2001). Resolution of conflicts involving many aircraft via semidefinite programming. *Journal of Guidance, Control, and Dynamics*, *24*(1), 79–86.

Gazi, V. (2006). Swarm aggregations using artificial potentials and sliding-mode control. *IEEE Transactions on Robotics*, *21*, 1208–1214.

Guy, S. J., Chhugani, J., Kim, C., Satish, N., Lin, M., Manocha, D., & Dubey, P. (2009). Clearpath: Highly parallel collision avoidance for multi-agent simulation. In *Proc. ACM SIGGRAPH Eurographics symposium on computer animation*.

Hill, J. C., Archibald, J. K., Stirling, W., & Frost, R. L. (2005). A multi-agent system architecture for distributed air traffic control. In *Proc. AIAA guidance, navigation and control conference*.

Hoffmann, G., & Tomlin, C. (2008). Decentralized cooperative collision avoidance for acceleration constrained vehicles. In *Proc. IEEE conference on decision and control*.

Hwang, I., & Tomlin, C. (2002). *Protocol-based conflict resolution for air traffic control* (Tech. Rep. SUDAAR-762). Stanford University.

Kuchar, J., & Yang, L. (2000). A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, *1*(4), 179–189.

Lalish, E. (2009). *Decentralized reactive collision avoidance*. PhD Thesis, University of Washington.

Lalish, E., & Morgansen, K. A. (2008). Decentralized reactive collision avoidance for multivehicle systems. In *Proc. IEEE conference on decision and control*.

Lalish, E., Morgansen, K. A., & Tsukamaki, T. (2006). Formation tracking control using virtual structures and deconfliction. In *Proc. IEEE conference on decision and control*.

Lalish, E., Morgansen, K. A., & Tsukamaki, T. (2008). Decentralized reactive collision avoidance for multiple unicycle-type vehicles. In *Proc. IEEE American control conference*.

Leitmann, G., & Skowronski, J. (1977). Avoidance control. *Journal of Optimization Theory and Applications*, *23*, 581–591.

Mastellone, S., Stipanovic, D. M., Graunke, C. R., Intlekofer, K. A., & Spong, M. W. (2008). Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments. *The International Journal of Robotics Research*, *27*(1), 107–126.

Melander, A. (2010). *Quadrotor implementation of the distributed reactive collision avoidance algorithm*. Master's Thesis, University of Washington.

Melander, A. P., Powel, N. D., Lalish, E., Morgansen, K. A., Jang, J. S., & Vian, J. (2010). Implementation of deconfliction in multivehicle autonomous systems. In *27th international congress of the aeronautical sciences*.

Pallottino, L., Feron, E., & Bicchi, A. (2002). Conflict resolution problems for air traffic management systems solved with mixed integer programming. *IEEE Transactions on Intelligent Transportation Systems*, *3*(1), 3–11.

Pallottino, L., Scordio, V. G., Bicchi, A., & Frazzoli, E. (2007). Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE Transactions on Robotics*, *23*(6), 1170–1183.

Pongpunwattana, A., & Rysdyk, R. (2004). Real-time planning for multiple autonomous vehicles in dynamic uncertain environments. *Journal of Aerospace Computing, Information, and Communication*, *1*, 580–604.

Reif, J. H., & Wang, H. (1999). Social potential fields: A distributed behavioral control for autonomous robots. *Robotics and Autonomous Systems*, *27*(3), 171–194.

Roussos, G. P., Dimarogonas, D. V., & Kyriakopoulos, K. J. (2008). 3D navigation and collision avoidance for a non-holonomic vehicle. In *Proc. IEEE American control conference*.

Skjetne, R., Moi, S., & Fossen, T. I. (2002). Nonlinear formation control of marine craft. In *Proc. IEEE conference on decision and control*.

Skogestad, S., & Postlethwaite, I. (2005). *Multivariable feedback control*. New York: Wiley.

Stanley, A. (2005). Flight path deconfliction of autonomous uavs. In *Infotech@Aerospace*.

Wang, S., & Schaub, H. (2008). Spacecraft collision avoidance using coulomb forces with separation distance and rate feedback. *Journal of Guidance, Control, and Dynamics*, *31*, 740–750.

Emmett Lalish received his Ph.D. in Controls in 2009 from the Department of Aeronautics and Astronautics at the University of Washington. He works at Moiré Inc., a small consulting firm, where he designs unmanned vehicles and their associated controllers. His interests include aircraft design, multidisciplinary optimization, nonlinear control and provably safe autonomy.

Kristi A. Morgansen received the B.S. (summa cum laude) and the M.S. in Mechanical Engineering from Boston University, Boston, MA, respectively in 1993 and 1994, and the S.M. in Applied Mathematics and Ph.D. in Engineering Sciences respectively in 1996 and 1999 from Harvard University, Cambridge, MA. After receiving the Ph.D. degree, she was first a postdoctoral scholar then a senior research fellow in Control and Dynamical Systems and Mechanical Engineering at the California Institute of Technology, Pasadena, CA. In August 2002, she joined the faculty of the Department of Aeronautics and Astronautics at the University of Washington where she is currently an Associate Professor with tenure. Her research interests include nonlinear and coordinated control systems, bioinspired sensing and actuation, fin-based propulsive methods, control of coordinated systems with communication constraints, vision-based sensing for state estimation, and development of integrated human and autonomous multivehicle systems. From 2002 to 2007, Professor Morgansen held the chaired position of Clare Boothe Luce Assistant Professor of Engineering at the University of Washington. She received a National Science Foundation (NSF) CAREER Award in 2003 and the 2010 O. Hugo Schuck Award for Best Paper in the Theory Category in the 2009 American Control Conference.