# Distributed Reactive Collision Avoidance for Multivehicle Systems

Emmett Lalish and Kristi A. Morgansen

***Abstract*— The focus of this paper is 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.**

## I. Introduction

As multi-vehicle autonomous systems are studied and implemented, the issue of conflict resolution becomes an increasingly important point. From mobile robots performing a cooperative search to air traffic control for unmanned aerial vehicles (UAVs), collision avoidance is of utmost importance for safety. The Distributed Reactive Collision Avoidance (DRCA) algorithm presented here is an extension of the previous work in [1], which performed collision avoidance for a planar group of $n$ variable-speed unicycles.

In order to capture the essential dynamics of a wider range of vehicles than in earlier work, this implementation uses three-dimensional double-integrator dynamics with arbitrary control (acceleration) saturation. Proper choice of saturation limits allow these simple dynamics to model even nonholonomic systems such as unicycles (in this way the results presented in [1] are still a special case of this more general algorithm). Vehicles with more complex dynamics (like aircraft, which must bank to turn) are accounted for in the robustness analysis, which details how to choose gains in order to account for filtering lags, time delays, and other unmodeled dynamics.

The motivation for this work comes from applications such as automating air traffic control and shipping traffic, where collisions must be avoided at all costs, and the amount of acceleration available compared to the speed is relatively small. Therefore, the point of this work is to guarantee that collisions will be avoided while taking into account the limited control authority available. Additionally, a centralized solution should be avoided so that a central node is not needed and so that the solution can be scaled to many vehicles by distributing the computational load across all of the vehicles involved.

This work supported by AFOSR grant FA9550-07-1-0528.

E. Lalish is with Moiré Inc., Issaquah, WA. emmett@moireinc.com.

K. A. Morgansen is with the Department of Aeronautics and Astronautics, University of Washington, Box 352400, Seattle, WA 98195-2400. morgansen@aa.washington.edu.

A wide variety of collision avoidance algorithms exist in the literature, of which an overview of some can be found in [2]. Why then do we need another one? The answer is that certain important aspects are lacking from each existing algorithm. Many existing collision avoidance algorithms do not actually guarantee collision avoidance [3], [4], [5], since for some applications (e.g. mobile robots) collisions will not necessarily destroy the vehicles and a heuristic approach will suffice.

Other algorithms guarantee avoidance without restricting the maximum acceleration available [6], [7], [8], which isn't really a guarantee at all. The reason is that any two vehicles with finite acceleration will have a space of initial conditions for which collision avoidance is impossible (for instance when they are approaching one another at relatively high speed). Therefore, to have a valid collision avoidance proof, one must also have a restriction on the initial conditions. This restriction will be related to quantities like breaking distance and turning radius, which may be negligible in some small-scale systems, but are fundamental to large-scale systems. Of the approaches that do guarantee avoidance in the presence of control limitations, many can only do so for a finite number of vehicles (usually two or three) [9], [10]. The work presented here accounts for $n$ vehicles simultaneously.

The final distinction comes with the degree of centralization. Several optimization approaches to collision avoidance [11], [12] yield strong results, however they are completely centralized and scalability is a major issue (even $O(n^3)$ is not great, and there is no guarantee of even this degree of scalability). On the other end of the scale is [13], which is completely decentralized, as in each vehicle needs only the states of vehicles within a given distance. It works by reserving an area the size of the minimum vehicle orbit and not letting any of those potential orbits overlap. While that algorithm may be ideal for mobile robot applications, its performance is fundamentally poor for vehicles with a large radius of curvature compared to their physical size.

The D in DRCA stands for distributed (instead of decentralized as in [1], which is not strictly correct), since the states of all other vehicles are required (not just nearest neighbors), but the computation is spread over the group. Each vehicle only needs to account for its own interactions, so the computations scale as $O(n)$ on each vehicle (making the overall computation $O(n^2)$).

This paper is organized as follows. Section II gives the problem statement and introduces definitions and notation used throughout the paper. The DRCA algorithm is described in Section III, while Section IV contains a robustness analysis thereof. Conclusions and future work are in Section V.

## II. 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).

### A. 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 do not affect the underlying concepts of conflict and collision. In this way, many different vehicle models work equivalently with this approach. To simplify the math, a simple vehicle model will be used for most of the following analysis: a 3D double integrator, which for the $i^{\text{th}}$ vehicle is given by

$$\frac{d}{dt}\begin{bmatrix}\mathbf{r}_i\\ \mathbf{v}_i\\ \Theta_i\end{bmatrix} = \begin{bmatrix}\mathbf{v}_i\\ \mathbf{u}_i\\ \Omega_i\Theta_i\end{bmatrix}, \tag{1}$$

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 $\Omega$ 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.

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 so it can be useful to tie this local coordinate frame to the actual body coordinates of the vehicle.

We constrain the input by use of an arbitrarily varying constraint set, $\mathbf{u}_i \in \mathcal{C}_i$. The only requirement is that $\mathcal{C}_i$ always contain the origin. A simple example of an input constaint set that limits maximum acceleration and velocity is

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

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

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

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. Let us model a vehicle which can go 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

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

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, $\mathcal{R}$ can be defined by

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

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 a holonomic model to a nonholonomic one, largely because of differences in controllability. However, controllability is not essential to the DRCA algorithm since orientations are arbitrary and only position and velocity matter. The DRCA algorithm is designed to use any control authority available, but it does not require the state space to be locally accessible.

### B. Definitions

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 $\dot{\tilde{\mathbf{v}}}_{ij} = \mathbf{u}_i - \mathbf{u}_j$.

In order to avoid collisions, first a strict definition of collision is necessary. These vehicles are modeled as nonholonomic point masses, however real vehicles have finite size. To account for this volume, a minimum allowed separation distance is included in the definition of a collision. This minimum distance could be, for example, the five nautical

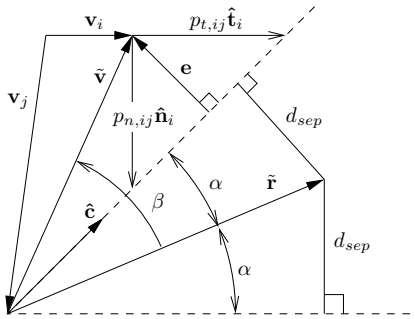


Fig. 1. Geometry of the $\mathbf{e}$ and $\hat{\mathbf{c}}$ vectors, as seen in an $\tilde{\mathbf{r}}$-$\tilde{\mathbf{v}}$ section through the 3D collision cone (dotted lines). The conflict measures $p_t$ and $p_n$ are shown, but $\hat{\mathbf{b}}$ points into the page, so $p_b$ is infinite ($\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$ in this example).

mile separation between aircraft required by the FAA or the sum of the radii of two mobile robots.

*Definition 1 (Collision):* A collision occurs between two vehicles when

$$\|\mathbf{r}\| < d_{sep}, \tag{5}$$

where $d_{sep}$ is the minimum allowed separation distance between the vehicles' centers. This distance can be different for each pair of vehicles in order to account for heterogeneity in the system.

For two vehicles not in a collision, the next question is whether they will collide if they remain at their present velocities. This first order prediction of a collision will be called a conflict.

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

$$\min_{t>0} \|\mathbf{r}(t)\| < d_{sep}. \tag{6}$$

*Lemma 1:* A necessary and sufficient condition for there to be no conflict is $|\beta| \geq \alpha$, where $\beta = \angle\mathbf{v} - \angle\mathbf{r}$ and $\alpha = \arcsin\left(\frac{d_{sep}}{\|\mathbf{r}\|}\right)$.

This lemma is proven in [14]. The angle $\alpha$ denotes the half-width of the collision cone, similar to [15], [11], [10], and is described geometrically in Fig. 1.

## III. DRCA Algorithm

The DRCA algorithm uses a two-step process: a deconfliction maneuver and a deconfliction maintenance phase. The deconfliction maneuver is designed to bring the vehicles to a conflict-free state. One basic deconfliction maneuver was presented in [1], in which each vehicle turns ($\mathbf{u}^\mathsf{T}\mathbf{v} = 0$) until the system reaches a conflict-free state. This maneuver can still be applied to this system, but better maneuvers have been developed in the meanwhile and will be presented in a separate publication. This paper will instead focus on the deconfliction maintenance controller.

Once a group of vehicles is conflict-free, the deconfliction maintenance controller will keep them that way. This controller allows each vehicle to use its desired control input unless that input would cause the vehicle to come into conflict

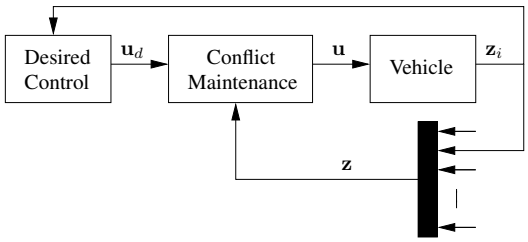


Fig. 2. Block diagram of the system when using the deconfliction maintenance controller. The vector $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]^\mathsf{T}$ and $\mathbf{z}_i = [\mathbf{r}_i, \mathbf{v}_i]^\mathsf{T}$. Note that the deconfliction maintenance block acts as a type of saturation on the desired control, $\mathbf{u}_d$.

with another vehicle. This algorithm and corresponding proof mirror [1], but extend it work in three dimensions and for the more general vehicle model (1). A basic block diagram of this setup is shown in Fig. 2.

In order to smoothly transition from the desired control to 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 $\alpha$ around a vector $\mathbf{q} = \tilde{\mathbf{r}} \times \tilde{\mathbf{v}}$ and normalizing:

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

Next, construct a normal vector, $\mathbf{e}$, from the collision cone to the relative velocity vector, $\tilde{\mathbf{v}}$ (see Fig. 1). If $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}} > 0$, then $\mathbf{e} = (I - \hat{\mathbf{c}}\hat{\mathbf{c}}^\mathsf{T})\tilde{\mathbf{v}}$, 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 $\mathbf{e} = \tilde{\mathbf{v}}$. Therefore:

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

In order to combine the effects of multiple collision cones, it is helpful to decompose the system into three component directions and analyze those directions 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 $\tilde{v}$ 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

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

which are described graphically in Fig. 1.

Define $\epsilon_t, \epsilon_n, \epsilon_b > 0$ as thresholds such that when $|p_t| > \epsilon_t$, 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.

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

and likewise for $p_n$ and $p_b$. Note that by definition $0 < p^{\pm} \leq \epsilon$. To simplify notation, in any case where a relation holds in 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

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

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

$$\begin{aligned} P_1 = (0, 0, 0) \qquad P_2 = (\epsilon, 0, u_{min}) \\ P_3 = (0, \epsilon, u_{max}) \qquad P_4 = (\epsilon, \epsilon, u_d). \end{aligned}$$

An example of this control function is shown in Fig. 3. 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 $\mathbf{u} \in \mathcal{R}$. Then the saturation function $S$ will give the final resultant control vector, which will be in $\mathcal{C}$. The octant-preserving nature of $S$ will ensure this final saturation step does not violate the requirements for the proof of collision avoidance.

A more intuitive way to choose values for $\epsilon$ is to relate it to a gain-like parameter,

$$k = \frac{u_{max} - u_{min}}{\epsilon}.$$

Note that $k$ has units of inverse seconds. Also, one can see that the magnitude of the gradient of the control function will always be less than or equal to $k$, regardless of the desired control.

*Theorem 1:* The deconfliction maintenance controller described above, when implemented on $n$ vehicles with dynamics (1) and inputs constrained by $\mathcal{C}_i$, 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. 1):

$$m = \begin{cases} \|\tilde{\mathbf{v}}\|, & \hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} \leq 0 \\ \|\tilde{\mathbf{v}}\| \sin(|\beta| - \alpha), & \hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0. \end{cases} \tag{12}$$

Note that $m$ is negative during conflict and positive outside of conflict.

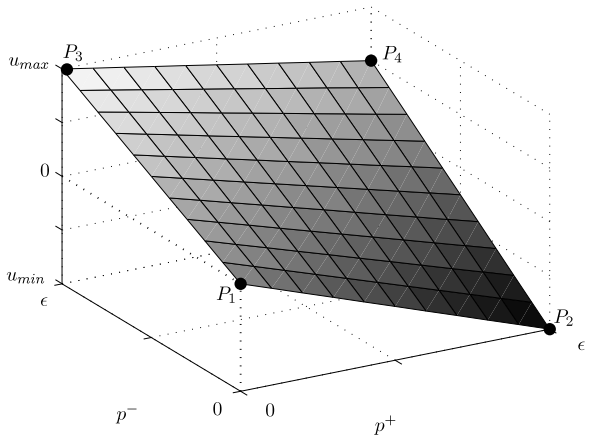


Fig. 3. Example of the control function, $F$. Note that $P_4$ moves up and down with changing $u_d$.

To ensure that a conflicted state is never reached (i.e. $m$ is always greater than zero), it is sufficient to show that

$$\lim_{m \to 0^+} \dot{m} \geq 0, \tag{13}$$

for every pair of vehicles. This condition implies that as the boundary of a collision cone approaches, it will either stop approaching or recede before a conflict is formed. The fact that this limit 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 (13) is satisfied, then $m > 0$ for all time, ensuring that $m \neq 0$.

The rest of this proof closely follows the two-dimensional version in [1], so most of the details will be skipped here for the sake of brevity. For $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} \leq 0$,

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

where $\hat{\mathbf{e}} = \frac{\mathbf{e}}{\|\mathbf{e}\|}$.

For $\hat{\mathbf{c}}^{\mathsf{T}}\tilde{\mathbf{v}} > 0$, the derivative of (12) becomes

$$\dot{m} = \sin(|\beta| - \alpha)\frac{d\|\tilde{\mathbf{v}}\|}{dt} + \|\tilde{\mathbf{v}}\|\cos(|\beta| - \alpha)\frac{d}{dt}(|\beta| - \alpha). \tag{15}$$

The derivative exists because $m > 0$ implies $\|\tilde{\mathbf{v}}\| \neq 0$, $|\beta| \geq \alpha > 0$, and $\|\tilde{\mathbf{r}}\| \geq d_{sep} > 0$. Taking these derivatives and simplifying (15) yields

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

This expression can be further simplified by recognizing that

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

Therefore

$$\dot{m} = \hat{\mathbf{e}}^{\mathsf{T}}\dot{\tilde{\mathbf{v}}} + m\frac{\|\tilde{\mathbf{v}}\|\cos(|\beta| - \alpha)}{\|\tilde{\mathbf{r}}\|\cos\alpha}. \tag{16}$$

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

$$\dot{m} \geq \hat{\mathbf{e}}^\mathsf{T}\dot{\tilde{\mathbf{v}}} \tag{17}$$

for any value of $\hat{\mathbf{c}}^\mathsf{T}\tilde{\mathbf{v}}$.

Expanding $\mathbf{u}_i$ into its components yields $\mathbf{u}_i = [u_{t_i}\hat{\mathbf{t}}_i + u_{n_i}\hat{\mathbf{n}}_i + u_{b_i}\hat{\mathbf{b}}_i]$. Following the logic in [1], (17) can be rewritten as (using the $ij$ notation again briefly for clarity)

$$\dot{m}_{ij} \geq m_{ij}\left(\frac{u_{t_i}}{p_{t,ij}} + \frac{u_{n_i}}{p_{n,ij}} + \frac{u_{b_i}}{p_{b,ij}} + \frac{u_{t_j}}{p_{t,ji}} + \frac{u_{n_j}}{p_{n,ji}} + \frac{u_{b_j}}{p_{b,ji}}\right). \tag{18}$$

As long as the controller ensures that $u_{t_i}$ has the same sign as $p_{t,ij}$, etc. then $\dot{m} \geq 0$ 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

$$\begin{aligned} &\lim_{p_{t_i}^+\to 0^+} u_{t_i} \geq 0, \quad \lim_{p_{n_i}^+\to 0^+} u_{n_i} \geq 0, \quad \lim_{p_{b_i}^+\to 0^+} u_{b_i} \geq 0, \\ &\lim_{p_{t_i}^-\to 0^+} u_{t_i} \leq 0, \quad \lim_{p_{n_i}^-\to 0^+} u_{n_i} \leq 0, \quad \lim_{p_{b_i}^-\to 0^+} u_{b_i} \leq 0, \end{aligned} \tag{19}$$

also ensures that

$$\lim_{m_{ij}\to 0^+} \dot{m}_{ij} \geq 0,$$

guaranteeing the system cannot enter a conflicted state.

The control function used in this implementation (10) satisfies (19), 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 $\mathcal{R}$ 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 $\mathcal{C}$ without affecting the guarantee. The key for this saturation to work is that $S$ preserves the octant of $\mathbf{u}$, such that (19) is still satisfied.

## IV. Robustness Analysis

Robustness for this system can be split into two areas: 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.

### A. Robust Stability

As complex as this control algorithm may appear, it can still be analyzed for robustness using linear tools. The deconfliction maneuver is a feed-forward control, so robust stability does not apply, since the input is static. 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 2:* Let $k_{ij} = k_{t_i}+k_{n_i}+k_{b_i}+k_{t_j}+k_{n_j}+k_{b_j}$. The system is stable in the presence of a complex multiplicative uncertainty bounded by $w(j\omega)$ if $k_{ij}$ satisfies

$$\frac{k_{ij}}{\sqrt{k_{ij}^2+\omega^2}} < \frac{1}{|w(j\omega)|} \quad \forall\omega. \tag{20}$$

*Proof:* Recalling (14), (16), and (18), one has that

$$\dot{m}_{ij} = m_{ij}\left(\frac{u_{t_i}}{p_{t,ij}} + \frac{u_{n_i}}{p_{n,ij}} + \frac{u_{b_i}}{p_{b,ij}} + \frac{u_{t_j}}{p_{t,ji}} + \frac{u_{n_j}}{p_{n,ji}} + \frac{u_{b_j}}{p_{b,ji}} + h_{ij}\right), \tag{21}$$

where $h_{ij}$ is a positive quantity given by

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

The worst case for robust stability is when the feedback gain is the largest (most negative), and one can see by looking at Fig. 3 that the largest gain in terms of $p^+$ will occur when $p^- = \epsilon$ 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,

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

Applying to (21) yields

$$\dot{m}_{ij} = -(k_{ij}-h_{ij})\,m_{ij} + \hat{\mathbf{e}}^\mathsf{T}\Theta_i\mathbf{u}_{i,max} + \hat{\mathbf{e}}^\mathsf{T}\Theta_j\mathbf{u}_{j,max}, \tag{22}$$

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 an $m$ 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

$$T(s) = \frac{L}{1+L} = \frac{k_{ij} - h_{ij}}{s + k_{ij} - h_{ij}}. \tag{23}$$

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

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

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 in [16]. From (23),

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

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 (20) 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 2:* The system is stable in the presence of a pure time delay no greater than $\tau$, if

$$k < \frac{1.47}{\tau}. \tag{24}$$

*Proof:* A multiplicative uncertainty, $w$, that bounds this time delay is given in [16]:

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

If the limiting case is when $\omega \geq \pi/\tau$, then (20) becomes

$$\frac{k}{\sqrt{k^2 + \omega^2}} < \frac{1}{2}$$
$$k < \frac{\omega}{\sqrt{3}} < \frac{\pi}{\tau\sqrt{3}}.$$

If the other case is limiting, then

$$\frac{k}{\sqrt{k^2 + \omega^2}} < \frac{1}{|1 - e^{-j\omega\tau}|}$$
$$\frac{k}{\sqrt{k^2 + \omega^2}} < \frac{1}{\sqrt{2 - 2\cos(\omega\tau)}}$$
$$k^2\left(1 - 2\cos(\omega\tau)\right) < \omega^2$$
$$k\tau < \frac{\omega\tau}{\sqrt{1 - 2\cos(\omega\tau)}}.$$

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 indeed limiting because $1.4775 < \pi/\sqrt{3}$. Therefore (24) is sufficient to guarantee stability in the presence of the time delay $\tau$. ■

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.

### *B. Robust Avoidance*

Another possible problem for this system is a vehicle which does not cooperate, i.e. does not satisfy (19). This 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 (19) 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 easy solution. However, for a two-vehicle system with a single adversary, the DRCA algorithm still provides a guarantee.

*Theorem 3:* For a two-vehicle system in which the vehicle's dynamics are restricted by (3), and 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

$$u_{1,max} \geq \frac{u_{2,max} v_{1,max}\sqrt{3}}{\sqrt{v_{1,max}^2 - v_{2,max}^2}}, \tag{25}$$

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

*Proof:* The choice of the control function (10) ensures that at the edge of a conflict, not only does vehicle one not approach the conflict any further, but it actually applies its maximum control authority in the opposite direction.

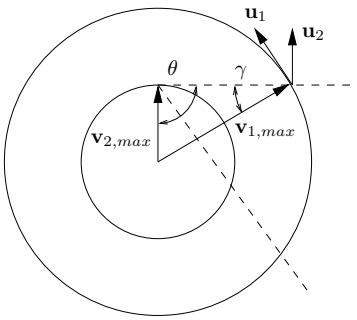


Fig. 4. Geometry of the worst situation for avoiding an adversary. The dotted lines represent the collision cone and the circles represent the set of allowable velocity vectors for the two vehicles.

Rederiving (22) for only vehicle one using deconfliction maintenance yields

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

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 get 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. 4, the worst angle for the collision cone to have is when $\gamma$ is maximized. The law of sines gives

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

so $\gamma$ 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 stay within $v_{1,max}$ is then

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

Substituting the maximum $\gamma$ from above and simplifying yields the bound (25). ■

## V. Conclusion

This work has developed a distributed control algorithm for deconflicting $n$ vehicles in three dimensions. The DRCA algorithm is reactive and so can easily be implemented real time on a wide variety of vehicles, including aircraft, ships, submarines and cars. Collision avoidance is guaranteed for a general $n$-vehicle system once a conflict-free state is reached, even in the case of arbitrarily small control authority. The DRCA algorithm allows the vehicles to follow changing desired controls so long as safety is not sacrificed. This algorithm has been shown to be robust to time delay and a variety of other unmodeled dynamics, and can even successfully evade an adversarial vehicle given certain bounds on the performance of the two vehicles.

The primary discussion omitted from this work is the deconfliction maneuver necessary to bring the system to a conflict-free state. Significant progress has been made in this area also, but must be discussed in a separate publication due to length constraints. That discussion will also include a method for adding and removing vehicles from the group. Finally, a discussion of liveness (guaranteeing the system does not become deadlocked), and a corresponding proof that the vehicles can get to where they are going in finite time will also be included.

## References


[1] E. Lalish and K. A. Morgansen, "Decentralized reactive collision avoidance for multivehicle systems," in *Proc. IEEE Conference on Decision and Control*, 2008.

[2] J. Kuchar and L. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intellingent Transportation Systems*, vol. 1, no. 4, pp. 179–189, 2000.

[3] M. Eby and W. Kelly, "Free flight separation assurance using distributed algorithms," in *Proc. IEEE Aerospace Conference*, 1999, pp. 429–441.

[4] J. C. Hill, J. K. Archibald, W. Stirling, and R. L. Frost, "A multi-agent system architecture for distributed air traffic control," in *Proc. AIAA Guidance, Navigation and Control Conference*, 2005.

[5] E. Lalish, K. A. Morgansen, and T. Tsukamaki, "Formation tracking control using virtual structures and deconfliction," in *Proc. IEEE Conference on Decision and Control*, 2006.

[6] J. Kosecka, C. Tomlin, G. Pappas, and S. Sastry, "Generation of conflict resolution maneuvers for air traffic management," in *Proc. International Conference of Intelligent Robotic Systems*, 1997, pp. 1598–1603.

[7] S. Mastellone, D. M. Stipanovic, C. R. Graunke, K. A. Intlekofer, and M. W. Spong, "Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments," *International Journal of Robotics Research*, vol. 27, no. 1, pp. 107–126, 2008.

[8] G. P. Roussos, D. V. Dimarogonas, and K. J. Kyriakopoulos, "3D navigation and collision avoidance for a non-holonomic vehicle," in *Proc. IEEE American Control Conference*, 2008.

[9] C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution for air traffic management: a study in multiagent hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 509–521, April 1998.

[10] C. Carbone, U. Ciniglio, F. Corraro, and L. Luongo, "A novel 3D geometric algorithm for aircraft autonomous collision avoidance," in *Proc. IEEE Conference on Decision and Control*, 2006.

[11] E. Frazzoli, Z. Mao, J. Oh, and E. Feron, "Resolution of conflicts involving many aircraft via semidefinite programming," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, pp. 79–86, 2001.

[12] L. Pallottino, E. Feron, and A. Bicchi, "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 3–11, March 2002.

[13] L. Pallottino, V. G. Scordio, A. Bicchi, and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multivehicle systems," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1170–1183, 2007.

[14] E. Lalish, K. A. Morgansen, and T. Tsukamaki, "Decentralized reactive collision avoidance for multiple unicycle-type vehicles," in *Proc. IEEE American Control Conference*, 2008.

[15] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 28, no. 5, pp. 562–574, September 1998.

[16] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control*. Wiley, 2005.