AIAA Guidance, Navigation, and Control Conference and Exhibit
16 - 19 August 2004, Providence, Rhode Island

AIAA 2004-5141

Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees

Tom Schouwenaars*, Jonathan How†, and Eric Feron‡
Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139

This paper presents a framework for provably safe decentralized trajectory planning of multiple (autonomous) aircraft. Each aircraft plans its trajectory individually using a receding horizon strategy based on mixed integer linear programming (MILP). A constrained, inertial, first-order linear model is used to capture the dynamics and kinematics of the vehicle. Safety is guaranteed by maintaining, at each time step, a dynamically feasible trajectory for each aircraft that terminates in a loiter pattern. Conflicts between multiple aircraft are resolved in a sequential, decentralized fashion, in which each aircraft takes into account the latest trajectory and loiter pattern of the other aircraft. Besides maintaining feasibility, if the problem is too complex to be solved within the time constraints of a real-time system, this approach also provides an *a priori* safe rescue solution consisting of the previous trajectories and individual loiter patterns. Several examples of conflict situations resolved by the proposed method are presented.

I. Introduction

The growing complexity of global air traffic has highlighted the shortcomings of the current air traffic control infrastructure. In the current system, air traffic controllers use predefined routes and standard procedures to ensure safe separation between the aircraft in their sector. By doing so, maintaining safety remains a manageable problem for the controller; however, the routes followed by the individual aircraft are often suboptimal. For example, the system disallows the aircraft to fly directly to the destination or to take advantage of favorable winds.[1] With the air space becoming more congested, delays are more frequent and accidents caused by the air traffic controller more likely to occur.

Therefore, in recent years, the concept of *Free Flight* has emerged, which allows pilots to choose their own routes, altitude and speed. Safe conflict detection and resolution schemes constitute the basis of such system, and have been a topic of active research. Automating these procedures reduces the risk of human errors and allows for optimization of the individual aircraft trajectories. Both *noncooperative* and *cooperative* conflict resolution methods have been proposed. In the noncooperative case, the aircraft involved in the conflict do not exchange information on their intentions and do not trust one another. Hence, a worst case approach is adopted. Examples include the work of Tomlin et al., in which a game-theoretic method is outlined.[1,2] Safe protocol-based maneuvers are derived by precomputing reachability sets[3,4] that account for the uncertainty in the actions of the other aircraft. Although, theoretically, the method can be applied to any number of aircraft, the computational requirements for more than three aircraft become prohibitive.

In cooperative conflict resolution schemes, the aircraft *do* exchange information on their positions and intentions. Within this class of methods, one can further distinguish between *centralized* and *decentralized* techniques, depending on whether the conflict is resolved by a central supervising controller or by each aircraft individually. In the former case, the position of each aircraft is known to the central controller who designs the individual trajectories for all aircraft, typically by solving one large optimization problem. Several methods with hard anti-collision constraints have been proposed, including approaches based on semidefinite programming,[5] nonlinear programming,[6] mixed integer linear programming,[7–9] mixed integer

*Research Assistant, Laboratory for Information and Decision Systems, toms@mit.edu, Student Member AIAA.
†Associate Professor, Space Systems Laboratory, jhow@mit.edu, Senior Member AIAA.
‡Associate Professor, Laboratory for Information and Decision Systems, feron@mit.edu, Associate Fellow AIAA.

nonlinear programming,[10] and variational analysis.[11] However, since the number of inter-vehicle combinations in the planning problem increases polynomially as $n(n-1)/2$ with the number of aircraft n involved in the conflict, the computation times of these methods typically scale exponentially.

By using a receding horizon approach, where the problem is solved over a limited time horizon that is shifted forward at each iteration, the computation time can be decreased. However, unless the problem is solved to completion at each iteration – which defeats the purpose of using a receding horizon–, safety is not guaranteed.[12] Namely, the algorithms may fail to provide a solution in future time steps, due to aircraft that are located beyond the surveillance and planning radius of the aircraft accounted for at the current time step. For instance, when the planning horizon is too short and the maximum turn rate relatively small, the aircraft might approach one another too closely before accounting for each other in their plans. As a result, they might not be able to turn away in time, which would translate into the optimization problem becoming infeasible.[13]

The scaling problem is also apparent in the field of unmanned aerial vehicles (UAV's). In applications where path planning and coordination of a large fleet of autonomous vehicles is required, centralized solutions quickly become computationally intractable. Moreover, in these applications, the planning problem typically needs to be resolved multiple times, as new information about the environment is often gathered while the mission unfolds. Thus, a *decentralized receding horizon* (or model predictive control) planning strategy seems a natural approach to solving the multi-vehicle trajectory generation problem. One such method is proposed in Ref. 14, where static obstacles and other moving agents are accounted for by potential functions. Although computationally attractive, the use of potential functions does not guarantee safety: obstacles and other vehicles are captured using soft "constraints" in the cost function. In Ref. 15, an alternative algorithm based on an iterative bargaining scheme is given. However, as the iteration might converge to an infeasible equilibrium, again only soft safety guarantees exist.

In this paper, we present a model predictive control framework for decentralized, *non-iterative* cooperative path planning of multiple aircraft with *hard* safety constraints. The novelty lies in the fact that safety is guaranteed by explicitly computing and maintaining a safe trajectory for each aircraft, without having to precompute an invariant set. Moreover, safety can be guaranteed for any number of interacting vehicles. This is achieved by ensuring *a priori* that each aircraft can always transition to a dynamically feasible loiter pattern. These loiter patterns act as safe "rescue" paths in case no feasible solution to a conflict can be found in time.

In our previous work,[12] the loiter principle was applied to the case of a single UAV navigating through a cluttered environment. In this paper, we extend the method to the case of multiple aircraft. Conflicts are resolved in a sequential, decentralized fashion in which each aircraft takes into account the latest trajectory and loiter pattern of the other when updating its own path. We model the problem as a *mixed integer linear program* (MILP) that needs to be solved in real-time. Applications of interest are commercial air traffic control and coordination of multiple UAV's.

The paper is organized as follows. Section II presents the problem formulation and Section III provides a high level description of the algorithm. Section IV then gives the detailed MILP formulation with the aircraft model and the loiter and avoidance constraints. In Section V, the framework is applied to some example scenarios, and Section VI concludes with topics for future research.

II. Problem Formulation

A. Receding Horizon Planning

The problem tackled in this paper is that of computing optimal trajectories in 2D for a set of (unmanned) aircraft while guaranteeing safety – to be defined later– at all times. Each aircraft individually computes its trajectory towards a destination waypoint, accounting for the intentions of the other aircraft. Since information on the latter is gathered online and changes as they update their own trajectories, each aircraft adopts a receding horizon planning strategy: a new segment of the total path towards the destination is computed at each time step by solving an optimization problem over a limited horizon of length T.[7,16] The cost function to be minimized can be a measure of time or fuel, or a more complex criterion such as visibility or risk. The solution to the optimization problem provides the trajectory points and corresponding input commands to the aircraft for the next T time steps. However, only the first of these input commands is actually implemented, and the process is repeated at the next time step. As such, new information about the state and actions of the (other) aircraft can be accounted for at each iteration.

We assume that the destination of each aircraft i consists of a final position $\boldsymbol{p}_{i,f}$ and a corresponding speed vector $\boldsymbol{v}_{i,f}$ with respect to an inertial coordinate frame. Together they constitute the final waypoint or state $\boldsymbol{x}_{i,f} = [\boldsymbol{p}'_{i,f}\ \boldsymbol{v}'_{i,f}]'$. This waypoint is typically an intermediate state along a flight plan or mission that was designed by a higher level decision unit. Given the state $\boldsymbol{x}_{i,t}$ at a certain time step t, the trajectory resulting from solving the path planning problem towards $\boldsymbol{x}_{i,f}$ consists of a sequence $\mathbf{x}_{i,t}$ of $(T+1)$ states $\boldsymbol{x}_{i,t+k}, k = 1 \dots T$, and a corresponding sequence $\mathbf{u}_{i,t}$ of T inputs $\boldsymbol{u}_{i,t+l}, l = 1 \dots (T-1)$:

$$\mathbf{x}_{i,t} = \begin{bmatrix} \boldsymbol{x}_{i,t} \\ \boldsymbol{x}_{i,t+1} \\ \vdots \\ \boldsymbol{x}_{i,t+T} \end{bmatrix},\ \mathbf{u}_{i,t} = \begin{bmatrix} \boldsymbol{u}_{i,t} \\ \boldsymbol{u}_{i,t+1} \\ \vdots \\ \boldsymbol{u}_{i,t+T-1} \end{bmatrix},\ \mathbf{p}_{i,t} = \begin{bmatrix} \boldsymbol{p}_{i,t} \\ \boldsymbol{p}_{i,t+1} \\ \vdots \\ \boldsymbol{p}_{i,t+T} \end{bmatrix},$$

where – for notational convenience later on– we denote the position part of the trajectory separately as $\mathbf{p}_{i,t}$.

The plan starting at time step t must be computed during time step $t-1$, i.e. when the aircraft is on its way to $\boldsymbol{x}_{i,t}$. The latter state is part of the previous plan, which we assume to be accurately tracked. As such, the aircraft will be in the predicted state $\boldsymbol{x}_{i,t}$ when the next plan is executed. We will make this assumption throughout the remainder of this paper. It implies that each aircraft can reliably assume that all other aircraft are exactly following their trajectories as planned. Including robustness to uncertainties in the latter is a topic of future research.

In what follows, we will denote the optimization problem for aircraft i that computes the trajectory starting at time step t as $\mathcal{M}_{i,t}$. It accounts for the (predicted) initial state $\boldsymbol{x}_{i,t}$, the destination waypoint $\boldsymbol{x}_{i,f}$, and constraints $\boldsymbol{x}_{i,k} \in \mathcal{X}_k$ on the states and $\boldsymbol{u}_{i,k} \in \mathcal{U}_k$ on the input commands. We call a solution *acceptable* if it is feasible and its cost lies within a predefined optimality gap. The details of how the optimization problem is set up are postponed until Section IV.

B. Loiter Pattern Principle

To clearly state what we mean by safe trajectory planning for an autonomous vehicle, we introduce the following definition:

Definition 1 (Vehicle Safety): *We say that a vehicle is in a* safe *state, if from that state, there exists a* known *dynamically feasible trajectory to a state or sequence of states that is obstacle- and collision-free, and in which the vehicle can remain for an indefinite period of time.* Safety *then implies that the vehicle is in a safe state at all times.*

For example, for a helicopter, safety can be ensured by maintaining a feasible trajectory to a hover state at an obstacle-free location. Similarly, for a rover, safety can be guaranteed by maintaining a feasible path towards a full stop. Aircraft, on the other hand, have minimum speed requirements, and therefore another type of "reachability" is required.

As was proposed in our previous work,[12] safety for a single aircraft i can be guaranteed by ensuring that the intermediate trajectory $\mathbf{x}_{i,t}$ computed at each time step t terminates in a loiter pattern, i.e. by constraining the last state $\boldsymbol{x}_{i,t+T}$ in the planning horizon to be an ingress state to a loiter pattern $\mathcal{L}_{i,t+T}$ that lies outside of any no-fly zones. As such, at the next time step $t+1$, a feasible solution to the trajectory optimization problem is always available *a priori*, namely, the remaining part of the previous trajectory $\mathbf{x}_{i,t}$ ending in the loiter pattern. Hence, in case at time step $t+1$, no *acceptable* solution to the trajectory optimization problem can be found within the timing limits of a hard real-time navigation system, the previous trajectory can be followed as a safe backup plan. If necessary, the latter can be tracked all the way from time step t to $t+T$, thus arriving at the ingress state $\boldsymbol{x}_{i,t+T}$ of the loiter pattern $\mathcal{L}_{i,t+T}$, in which the aircraft can remain for an indefinite period of time. Note that this safety principle assumes that the waypoint sequence $\mathbf{x}_{i,t}$ can be accurately tracked, which requires the plan to be *robust* to disturbances and plant uncertainties.

We can now specialize the definition of safety to the case of an aircraft:

Definition 2 (Aircraft Safety): *We say that an aircraft i is in a* safe *state $\boldsymbol{x}_{i,t}$, if from that state, there exists a* known *robust, dynamically feasible trajectory $\mathbf{x}_{i,t}$ ending in an obstacle- and collision-free loiter pattern $\mathcal{L}_{i,t+T}$.* Safety *then implies that the aircraft is in a safe state at all times.*

C. Conflict Description

The principle of maintaining a reachable loiter pattern is key to ensuring safety in case of an encounter between multiple aircraft. It will form the basis of our algorithm for safe trajectory planning and conflict resolution. For simplicity of exposition, we assume that all aircraft are identical and that their planning horizons are the same. Extending the framework to the more general case of different aircraft dynamics and unequal planning horizons can be done at the cost of a more complicated notation. We start with some more definitions:

Definition 3 (Conflict Zone): *We denote by* $\mathcal{R}_{i,t} \subset \Re^2$ *the subset of the inertial space that encompasses the area in which all dynamically feasible trajectories lie that start at* $\boldsymbol{x}_{i,t} = [\boldsymbol{p}'_{i,t}\ \boldsymbol{v}'_{i,t}]'$ *and end in feasible loiter patterns*: $\forall \mathbf{u}_{i,t} \in \mathcal{U} = \{\mathcal{U}_0, \ldots, \mathcal{U}_{T-1}\} : \mathbf{p}_{i,t} \cup \mathcal{L}_{i,t+T} \subset \mathcal{R}_{i,t}$. *We call* $\mathcal{R}_{i,t}$ *the* conflict zone *of aircraft* i *at time step* t.

As an initial condition (at $t = 0$) for the planning and conflict resolution algorithm, we now assume that none of the conflict zones $\mathcal{R}_{i,0}$ of the individual aircraft i ($i = 1 \ldots K$) in the air space overlap. As such, at their initial positions $\boldsymbol{p}_{i,0}$, all aircraft can safely plan their individual trajectories and loiter patterns without accounting for the other aircraft.

Assumption 1 (Initial Safety): *At t=0, we have* $\mathcal{R}_{1,0} \bigcap \mathcal{R}_{2,0} \ldots \bigcap \mathcal{R}_{K,0} = \emptyset$.

When the conflict zones of two or more aircraft start to overlap, however, the individually planned trajectories might intersect and lead to a collision. Therefore, as soon as an overlap is detected, the aircraft should account for each other's trajectories when updating their plans. We call this a *conflict*, and define it more formally as follows:

Definition 4 (Conflict): *We say that aircraft* i *is involved in a* conflict $\mathcal{C}_{ij,t}$ *with aircraft* $j \neq i$ *at time step* t, *if* $\mathcal{R}_{i,t} \cap \mathcal{R}_{j,t} \neq \emptyset$.

Since an aircraft i computes the trajectory that starts at time step t during step $t-1$, $\mathcal{R}_{i,t}$ needs to be determined at $t-1$ as well, based on the prediction of $\boldsymbol{x}_{i,t}$. To denote the set of aircraft that are then involved in a conflict with aircraft i at time step t, and for which avoidance constraints must be formulated to avoid collisions, we introduce the following notation:

Definition 5 (Avoidance Set): *We denote by* $\mathcal{J}_{i,t} \subset \{1 \ldots K\}$ *the subset of all aircraft* $j \in \{1 \ldots K\}, j \neq i$, *for which* $\mathcal{R}_{i,t} \cap \mathcal{R}_{j,t} \neq \emptyset$. *We call* $\mathcal{J}_{i,t}$ *the* avoidance set *of aircraft* i *at time step* t.

If two aircraft i and j are involved in a conflict $\mathcal{C}_{ij,t}$, however, their avoidance sets $\mathcal{J}_{i,t}$ and $\mathcal{J}_{j,t}$ are not necessarily the same: aircraft j can be involved in a conflict $\mathcal{C}_{jk,t}$ with another aircraft k, but $\mathcal{R}_{i,t} \cap \mathcal{R}_{k,t} = \emptyset$. Hence, to maintain safety, aircraft j must account for both i and k, whereas aircraft i only has to account for j. However, although they are not directly in conflict, the trajectory of i will still be influenced by that of k through the effect that k has on the trajectory of j. This indirect dependence plays a key role in maintaining safety during the decentralized planning algorithm: aircraft k has to come into play when the conflict $\mathcal{C}_{ij,t}$ between i and j is solved. We therefore introduce the following set:

Definition 6 (Conflict Set): *We denote by* $\mathcal{S}_{i,t} \subset \{1 \ldots K\}$ *the subset of all aircraft* $j \in \{1 \ldots K\}$ *for which there exists an aircraft sequence* $k_1, \ldots, k_S$ *such that* $\mathcal{R}_{i,t} \cap \mathcal{R}_{k_1,t} \neq \emptyset$, $\mathcal{R}_{k_1,t} \cap \mathcal{R}_{k_2,t} \neq \emptyset, \ldots, \mathcal{R}_{k_S,t} \cap \mathcal{R}_{j,t} \neq \emptyset$. *We call* $\mathcal{S}_{i,t}$ *the* conflict set *of aircraft* i *at time step* t.

Note that if $j \in \mathcal{S}_{i,t}$, we have $\mathcal{S}_{j,t} = \mathcal{S}_{i,t}$. Hence, we can define one set $\mathcal{S}_t \equiv \mathcal{S}_{i,t}$ that groups all aircraft that are connected through a chain of conflicts $\mathcal{C}_{ik_1,t}, \ldots, \mathcal{C}_{k_S j,t}$ at time step t. The aircraft in $\mathcal{S}_t$ and their individual conflicts $\mathcal{C}_{ij,t}$ form a connected graph. Any aircraft that do not belong to $\mathcal{S}_t$ will not come into play in resolving this *graph conflict* $\mathcal{C}_t$. Since two distinct graph conflicts are then fully decoupled, without loss of generality, we can restrict ourselves to solving the case of a single graph conflict $\mathcal{C}_t$.

D. Communication Requirements

To determine if a connecting conflict sequence exists between any two aircraft at any time, there needs to be a communication link between all aircraft in the air space at all times. Alternatively, all aircraft i only communicate with a central hub (either a *ground station* or *leader* aircraft) that keeps track of the positions

$\boldsymbol{p}_{i,t}$ and conflict zones $\mathcal{R}_{i,t}$ at all time steps, and determines the conflict set $\mathcal{S}_t$ from this information. Moreover, if desired, this central entity can also perform the task of computing the avoidance sets $\mathcal{J}_{i,t}$ and communicate these back to the respective aircraft i. An additional benefit is that it can maintain the clocks of all aircraft synchronized, which will be crucial in the conflict resolution part of the algorithm.

Assumption 2 (Central Hub): *We assume that a central hub $\mathcal{H}$ is present with which all aircraft $i = 1 \dots K$ communicate. It keeps the clocks of all aircraft synchronized, and determines the avoidance sets $\mathcal{J}_{i,t}$ and the conflict set $\mathcal{S}_t$.*

Although the existence of a central hub is not crucial to our safety framework, it simplifies the required communication infrastructure and the presentation of the algorithm. The actual trajectory optimization will still be done in a decentralized fashion, regardless of how information is exchanged between the aircraft. Decentralizing the communication itself is an ad hoc networking problem for which multiple techniques exist.[17,18]

To avoid collisions, aircraft i needs to account for the position sequence $\mathbf{p}_{j,t}$ of all aircraft $j \in \mathcal{J}_{i,t}$ when solving its trajectory optimization problem $\mathcal{M}_{i,t}$. Moreover, to maintain future safety, its loiter pattern $\mathcal{L}_{i,t+T}$ should not intersect with any of the loiter patterns $\mathcal{L}_{j,t+T}$ of the aircraft $j \in \mathcal{J}_{i,t}$. As will be shown in Section IV, in our MILP planning algorithm, it suffices to describe all $\mathcal{L}_{j,t+T}$ by the coordinates of a surrounding rectangle that is aligned with the inertial coordinate frame. The latter will then be considered as a no-fly zone by aircraft i.

To solve its trajectory optimization problem $\mathcal{M}_{i,t}$, aircraft i thus needs to obtain the latest trajectory information of all aircraft $j \in \mathcal{J}_{i,t}$, and vice versa, the latter need to know the trajectory of i. Hence, either communication links between all aircraft in $\mathcal{J}_{i,t}$ and aircraft i need to be set up, or the information can be distributed via the central hub. We now introduce the following notation to capture this trajectory information:

Definition 7 (Plan): *We denote by $\mathcal{P}_{i,t}$ the sequence $\mathbf{p}_{i,t}$ of trajectory points starting at time step t and the coordinates of a rectangle that is aligned with the inertial coordinate frame and surrounds the loiter pattern $\mathcal{L}_{i,t+T}$ of aircraft i. We call $\mathcal{P}_{i,t}$ the* plan *of aircraft i at time step t.*

The plans $\mathcal{P}_{j,t}$ for all aircraft $j \in \mathcal{J}_{i,t}$ can then be included as avoidance constraints in the trajectory optimization problem $\mathcal{M}_{i,t}$. We will denote this as the problem $\mathcal{M}_{i,t}$ *s.t.* $\mathcal{P}_{j,t}, \forall j \in \mathcal{J}_{i,t}$.

A key step in our conflict resolution algorithm is that the aircraft in $\mathcal{S}_t$ need to decide on an order $\mathcal{O}_t$ in which each one updates its trajectory at time step $t-1$. Again, the easiest way to accomplish this is to let the central hub assign this order. Associated with it, is a sequence of non-overlapping time slots during which a particular aircraft must start and end its trajectory optimization:

Assumption 3 (Conflict Order): *We assume that the central hub $\mathcal{H}$ can determine an order $\mathcal{O}_t = \{ord(i), i \in \mathcal{S}_t\}$, and a corresponding sequence of non-overlapping time slots $\{[t_{ord(i),s}, t_{ord(i),f} + \Delta t_{comm}), i \in \mathcal{S}_t\}$ that give the allocated start time $t_{ord(i),s}$ and end time $t_{ord(i),f}$ during which the aircraft with order number $ord(i)$ should solve its trajectory optimization problem at time step $t-1$. Δt_{comm} is the time required to broadcast the plan $\mathcal{P}_{i,t}$ to the aircraft in $\mathcal{J}_{o_i,t}$.*

Since an aircraft might leave the conflict chain, or a new one might enter, the central hub must redetermine $\mathcal{S}_t$ and $\mathcal{O}_t$ at every time step $t-1$. Using the definitions from above, we can now formulate a decentralized optimization strategy for safe trajectory planning and conflict resolution.

III. Safe Trajectory Planning Algorithm

A. Algorithm

Starting at $t = 0$, at each following time step t, let all aircraft i in the air space ($i = 1 \dots K$) execute the following planning algorithm:

Start: Start at time t.

- **Step 1**: Predict the *next state* $\boldsymbol{x}_{i,t+1}$ and send it to the *central hub* $\mathcal{H}$. Next, receive the *avoidance set* $\mathcal{J}_{i,t+1}$, the *conflict order* $\mathcal{O}_{t+1}$, and $(t_{ord(i),s}, t_{ord(i),f})$ from $\mathcal{H}$.

- **Step 2a**: If $\mathcal{J}_{i,t+1} \neq \emptyset$, broadcast the *current plan* $\mathcal{P}_{i,t}$ to all aircraft in $\mathcal{J}_{i,t+1}$ and go to **Step 3**. Else, go to **Step 2b**.
- **Step 2b**: Solve $\mathcal{M}_{i,t+1}$. If an *acceptable solution* $\mathbf{x}_{i,t+1} \cup \mathcal{L}_{i,t+T+1}$ is found before time $t+1$, let the *new plan* $\mathcal{P}_{i,t+1} = \mathbf{p}_{i,t+1} \cup \mathcal{L}_{i,t+T+1}$. Else, let $\mathcal{P}_{i,t+1} = \mathcal{P}_{i,t} \setminus \boldsymbol{p}_{i,t}$. Go to **End**.
- **Step 3**: At time $t_{ord(i),s}$, solve $\mathcal{M}_{i,t+1}$ *s.t.* $\mathcal{P}_{j,t+1}, j \in \mathcal{J}_{i,t+1} : ord(j) < ord(i)$ *and s.t.* $\mathcal{P}_{k,t}, k \in \mathcal{J}_{i,t+1} : ord(k) > ord(i)$.
- **Step 4**: If an *acceptable solution* $\mathbf{x}_{i,t+1} \cup \mathcal{L}_{i,t+T+1}$ is found at $t_{ord(i),f}$, let the *new plan* $\mathcal{P}_{i,t+1} = \mathbf{p}_{i,t+1} \cup \mathcal{L}_{i,t+T+1}$. Else, let $\mathcal{P}_{i,t+1} = \mathcal{P}_{i,t} \setminus \boldsymbol{p}_{i,t}$.
- **Step 5**: During $(t_{ord(i),f}, t_{ord(i),f} + \Delta t_{comm})$, broadcast $\mathcal{P}_{i,t+1}$ to all aircraft $k \in \mathcal{J}_{i,t+1} : ord(k) > ord(i)$.

End: End by time $t+1$, and repeat.

By construction, this algorithm maintains safety for all aircraft: at each time step, there exists, *a priori*, a collision-free dynamically feasible trajectory for each aircraft. Namely, since all other aircraft accounted for the latest plan of the aircraft that is planning, the remaining part of its previous trajectory ending in a loiter pattern can always be used as a safe "backup" plan. We formalize this in the following theorem:

Theorem: *Given a conflict-free situation at time $t = 0$ (Assumption 1), the above planning algorithm will maintain safe trajectories (Definition 2) for all aircraft $i = 1 \dots K$ at all time steps.*

Proof: We prove the safety property by using a double induction argument over the sequence of time steps and over the sequence of aircraft within a time step. By Assumption 1, at $t = 0$, all aircraft are in a safe state (as defined in Definition 2) and $\bigcap_{i=1\dots K} \mathcal{R}_{i,0} = \emptyset$. Hence, by definition of $\mathcal{R}_{i,0}$, all aircraft i have non-intersecting plans at $t = 0$. They thus all have an *a priori* safe trajectory available at $t = 1$, namely, $\mathcal{P}_{i,0} \backslash \boldsymbol{p}_{i,0}$ ending in non-intersecting loiter patterns $\mathcal{L}_{i,T}$ at time step T, the length of the planning horizon.

Assume now that at time step $t = k$, all aircraft have safe plans $\mathcal{P}_{i,k}$. Before computing its next plan $\mathcal{P}_{i,k+1}$, a particular aircraft i will know whether its avoidance set $\mathcal{J}_{i,k+1}$ is empty or not by communicating with the central hub $\mathcal{H}$ (Step 1). In case $\mathcal{J}_{i,k+1} = \emptyset$, any feasible solution to $\mathcal{M}_{i,k+1}$ will be safe, since the conflict zone $\mathcal{R}_{i,k+1}$ does not intersect with that of any other aircraft. If an acceptable solution cannot be found in time, $\mathcal{P}_{i,k} \backslash \boldsymbol{p}_{i,k}$, i.e. the previous plan excluding the current state, is still valid and available as a safe backup plan at time step $k+1$.

In case $\mathcal{J}_{i,k+1} \neq \emptyset$, aircraft i will obtain its order number $ord(i) \in \{1 \dots |\mathcal{S}_{k+1}|\}$ from the central hub $\mathcal{H}$, where $|\mathcal{S}_{k+1}|$ is the cardinality of the conflict set. If $ord(i) = 1$, aircraft i is the first to update its trajectory at time step k. Given that all plans $\mathcal{P}_{i,k}$ were safe, a safe solution to $\mathcal{M}_{i,k+1}$ *s.t.* $\mathcal{P}_{j,k+1}, \forall j \in \mathcal{J}_{i,k+1}$ continues to exist, namely, $\mathcal{P}_{i,k} \backslash \boldsymbol{p}_{i,k}$. In the nominal case, $\mathcal{P}_{i,k+1}$ will differ from $\mathcal{P}_{i,k} \backslash \boldsymbol{p}_{i,k}$. Since the updated plan $\mathcal{P}_{i,k+1}$ is constrained to avoid the existing trajectories $\mathcal{P}_{j,k}, \forall j \in \mathcal{J}_{i,k+1}$, all other aircraft in the conflict set $\mathcal{S}_{k+1}$ remain in safe states.

Now, consider an aircraft i' with $ord(i') > 1$, and assume that it has a safe backup plan $\mathcal{P}_{i',k} \backslash \boldsymbol{p}_{i',k}$. It will account *i*) for the previous plans $\mathcal{P}_{j',k}$ for all following aircraft $j' \in \mathcal{J}_{i',k+1} : ord(j') > ord(i')$ and *ii*) for the new plans $\mathcal{P}_{l',k+1}$ of all prior aircraft $l' \in \mathcal{J}_{i',k+1} : ord(l') < ord(i')$. Given that a safe plan $\mathcal{P}_{i',k} \backslash \boldsymbol{p}_{i',k}$ for aircraft i' exists, the problem is feasible. Thus *i*) all aircraft j' with $ord(j') > ord(i')$ will still have safe backup plans $\mathcal{P}_{j',k} \backslash \boldsymbol{p}_{j',k}$ available when they update their paths, and *ii*) the new plans $\mathcal{P}_{l',k+1}$ of all aircraft l' with $ord(l') < ord(i')$ will remain safe. Hence, by induction over the sequence of aircraft, safety for all aircraft is maintained within time step k. As a result, each aircraft is in a safe state at the start of the next time step $k+1$. Thus, using induction once more, given the safety assumption at $t = 0$, safety for all aircraft is maintained over the sequence of time steps.

B. Remarks

1. Note that the algorithm is *not* a bargaining or convergence process: at a given time step, each aircraft contributes only once to the solution of the conflict. As presented here, the algorithm should cycle through the full conflict set $\mathcal{S}_{k+1}$ before the next time step, i.e. the time at which all aircraft reach the first state in their plans. However, the more aircraft that are involved in the conflict, the longer each aircraft will typically take to solve its trajectory optimization problem. On the other hand, the allocated computation

time scales inversely with the number of aircraft involved. Therefore, if necessary, for a particular cycle of the algorithm, the central hub can distribute the computation times over several time steps. As long as during this cycle, all other aircraft keep tracking the latest trajectories that they communicated, safety for all aircraft is maintained. This is only the case, however, if no new aircraft enter the conflict chain during that longer cycle. Since one iteration of the conflict resolution algorithm is now spread over more than one time step, this can be guaranteed by computing conflict sets for longer planning horizons in Step 1 of the algorithm. Alternatively, the central hub could command the new aircraft not to update its plan and to keep tracking its latest trajectory, while it communicates this latest plan to the other aircraft. The latter can then still include that plan as an avoidance constraint in their respective trajectory optimizations, without making the problem infeasible.

2. Worst-case, if too many aircraft are involved in the chain conflict, and the problem becomes too complex to be solved within the timing constraints of the algorithm, the aircraft will enter their (non-intersecting) loiter patterns. The central hub can then decide on subsequent subsets of the conflict set (including the possibility of one by one) and allocate as much time as needed to compute a way out of the loiter patterns.

3. It is natural to assume that the first aircraft in the conflict order is favored, since it can update its trajectory first. However, in our simulations this was not necessarily the case: we generally found that most aircraft have to make equal efforts (in terms of deviating from their nominal trajectories towards their destination) to resolve the conflict. Depending on the geometry of the individual trajectories, the first aircraft could even be more constrained than the last one in the order sequence. Moreover, by randomly changing the aircraft order at each time step, no single aircraft should have an *a priori* advantage over any other.

4. The algorithm is robust to communication failures. As soon as a broken communication link is detected, the aircraft involved should execute their backup plans: they should keep following the trajectory that was last broadcast and eventually enter their loiter pattern. The other aircraft can then keep accounting for those same plans when updating their trajectories.

5. The formulation of the actual trajectory optimization problem should account for the discrete time nature of the conflict resolution algorithm: its constraints must guarantee collision avoidance during the continuous transition between the discrete trajectory sample points.

IV. Mathematical Formulation

In this section, we cast the trajectory planning algorithm described above in a mathematical framework based on *mixed integer linear programming* (MILP). Although the model can readily be extended to account for changes in altitude, we restrict ourselves to the two dimensional case. Moreover, MILP is not the only trajectory optimization method which can be used to implement the safe planning algorithm.

A. Aircraft Model

1. Continuous Version

In our earlier work on path planning using MILP, an inertial double integrator model was used with additional constraints to capture bounds on speed and turn rate.[7,8] In this paper, we introduce an inertial velocity control model and alternative turn rate constraints that capture the behavior of an actual aircraft more accurately. As mentioned before, to simplify the notation, we assume that all aircraft behave according to the same dynamics and constraints.

In what follows, we drop the subscripts i and t indicating the aircraft that is planning and the current time step. Let $\boldsymbol{p} = (x, y)$ and $\boldsymbol{v} = (\dot{x}, \dot{y})$ denote the inertial position and velocity vector respectively. Let the inputs to the aircraft consist be an inertial reference speed vector $\boldsymbol{u} = (\dot{x}_{cmd}, \dot{y}_{cmd})$. We then use the following continuous, first-order velocity control model:

$$\begin{aligned} \ddot{x}(t) &= -\tfrac{1}{\tau}\dot{x}(t) + \tfrac{k}{\tau}\dot{x}_{cmd}(t) \\ \ddot{y}(t) &= -\tfrac{1}{\tau}\dot{y}(t) + \tfrac{k}{\tau}\dot{y}_{cmd}(t) \end{aligned} \tag{1}$$

in which τ is a time constant and k is a gain. In state space form, $\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t)$, we obtain:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \ddot{x}(t) \\ \ddot{y}(t) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{1}{\tau} & 0 \\ 0 & 0 & 0 & -\frac{1}{\tau} \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \\ \dot{x}(t) \\ \dot{y}(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{k}{\tau} & 0 \\ 0 & \frac{k}{\tau} \end{bmatrix} \begin{bmatrix} \dot{x}_{cmd}(t) \\ \dot{y}_{cmd}(t) \end{bmatrix} \tag{2}$$

Note that the model is symmetric in the x- and y-coordinate, which is corrected for by adding inequality constraints as follows. The magnitude of the reference and actual inertial velocity is bounded by a minimum required speed v_{min} and a maximum achievable speed v_{max}:

$$\begin{aligned} v_{min} \leq ||\boldsymbol{v}(t)|| \leq v_{max} \\ v_{min} \leq ||\boldsymbol{u}(t)|| \leq v_{max} \end{aligned} \tag{3}$$

Bounds on turn rate are captured by approximating the geometric profile that encompasses all inertial acceleration vectors that are dynamically feasible. We assume that the latter is a rectangular area delimited by the maximum forward and lateral accelerations $a_{fwd,max}$ and $a_{lat,max}$. Since it is heading dependent, the rectangle in which the inertial acceleration vector $\boldsymbol{a} = (\ddot{x}, \ddot{y})$ must lie should be aligned with the velocity vector at all times. This requirement can be modeled by the following set of inequalities, which are explained in more detail in Ref. 19:

$$\underline{\text{case 1}}: \quad a_{fwd,max} \leq a_{lat,max}$$

$$\begin{cases} ||\boldsymbol{a}(t) - \alpha\frac{\boldsymbol{v}(t)}{v_0}|| & \leq & \beta \\ ||\boldsymbol{a}(t) + \alpha\frac{\boldsymbol{v}(t)}{v_0}|| & \leq & \beta \\ ||\boldsymbol{a}(t)|| & \leq & a_{lat,max} \end{cases} \tag{4}$$

$$\underline{\text{case 2}}: \quad a_{fwd,max} > a_{lat,max}$$

$$\begin{cases} ||\boldsymbol{a}(t) - \alpha_\perp\frac{\boldsymbol{v}_\perp(t)}{v_0}|| & \leq & \beta_\perp \\ ||\boldsymbol{a}(t) + \alpha_\perp\frac{\boldsymbol{v}_\perp(t)}{v_0}|| & \leq & \beta_\perp \\ ||\boldsymbol{a}(t)|| & \leq & a_{fwd,max} \end{cases} \tag{5}$$

Here, $\boldsymbol{v}_\perp$ denotes the orthogonal complement of $\boldsymbol{v}$, v_0 indicates the magnitude of the current velocity and $\alpha, \beta, \alpha_\perp$ and $\beta_\perp$ are appropriate constants that are determined by the shape of the rectangular profile.

2. Discrete Version

To make use of the above dynamics in a MILP framework, we introduce a discrete time version of the model $(\mathbf{A}, \mathbf{B})$ and linear approximations of the inequality constraints. By introducing a sample time Δt and using the bilinear transform, we obtain the following constraints describing the discrete time vehicle dynamics:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \mathbf{A}_d\boldsymbol{x}_k + \mathbf{B}_d\boldsymbol{u}_k, \ k = 0, \ldots, T-1 \\ \boldsymbol{x}_0 &= [\boldsymbol{p}_0' \ \boldsymbol{v}_0']' \end{aligned} \tag{6}$$

with

$$\begin{aligned} \mathbf{A}_d &= \left(\mathbf{I} - \tfrac{\Delta t}{2}\mathbf{A}\right)^{-1}\left(\mathbf{I} + \tfrac{\Delta t}{2}\mathbf{A}\right) \\ \mathbf{B}_d &= \left(\mathbf{I} - \tfrac{\Delta t}{2}\mathbf{A}\right)^{-1}\mathbf{B}. \end{aligned} \tag{7}$$

The quadratic inequality constraints (3), (4) and (5) are approximated by the edges of an N-sided polygon. For the acceleration constraints (4), we obtain:

$\forall k \in [0, \ldots, T-1], \forall n \in [1, \ldots, N]:$

$$\left(\frac{\dot{x}_{k+1} - \dot{x}_k}{\Delta t} - \alpha\frac{\dot{x}_k}{v_0}\right)\sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{k+1} - \dot{y}_k}{\Delta t} - \alpha\frac{\dot{y}_k}{v_0}\right)\cos\left(\frac{2\pi n}{N}\right) \leq \beta \tag{8}$$

$$\left(\frac{\dot{x}_{k+1} - \dot{x}_k}{\Delta t} + \alpha\frac{\dot{x}_k}{v_0}\right)\sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{k+1} - \dot{y}_k}{\Delta t} + \alpha\frac{\dot{y}_k}{v_0}\right)\cos\left(\frac{2\pi n}{N}\right) \leq \beta \tag{9}$$

$$\left(\frac{\dot{x}_{k+1} - \dot{x}_k}{\Delta t}\right)\sin\left(\frac{2\pi n}{N}\right) + \left(\frac{\dot{y}_{k+1} - \dot{y}_k}{\Delta t}\right)\cos\left(\frac{2\pi n}{N}\right) \leq a_{lat,max} \tag{10}$$

and similarly for (5). To ensure a smooth input profile, these inequalities should also be formulated in terms of the reference velocity commands $(\dot{x}_{cmd,k},\ \dot{y}_{cmd,k})$ (for $k = 0, \ldots, T-1$). In the same way, the upper bound of the velocity constraints (3) translates into:

$\forall k \in [0, \ldots, T], \forall n \in [1, \ldots, N]$:

$$\dot{x}_k \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_k \cos\left(\frac{2\pi n}{N}\right) \leq v_{max} \tag{11}$$

and similarly for the reference velocity commands. The minimum speed requirement in (3) can be handled by ensuring that the speed vector lies *outside* an N-sided polygon rather than inside, as was the case for the previous constraints. By introducing binary variables, we can capture this non-convex constraint as follows:

$\forall k \in [0, \ldots, T], \forall n \in [1, \ldots, N]$:

$$\begin{aligned} \dot{x}_k \sin\left(\frac{2\pi n}{N}\right) + \dot{y}_k \cos\left(\frac{2\pi n}{N}\right) &\geq v_{min} - M c_{kn} \\ \sum_{n=1}^{N} c_{kn} &\leq N - 1 \\ c_{kn} &\in \{0, 1\} \end{aligned} \tag{12}$$

where M is a sufficiently large number. Again, these constraints should also be formulated for the reference commands.

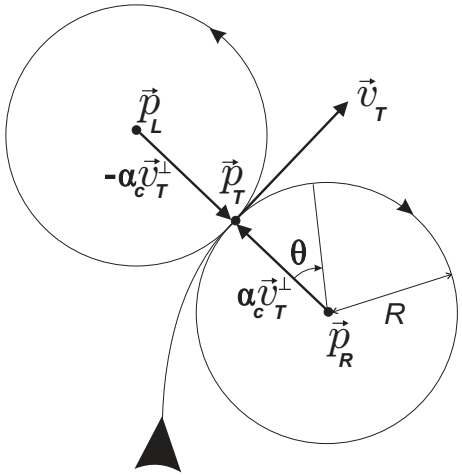


Figure 1. Safe trajectory ending in either a left or right turning loiter circle.

B. Loiter Circles

As discussed in Section II, the trajectory at each iteration is constrained to terminate in a loiter pattern. Here, we choose this to be a left or right turning circle (see Figure 1), where the turn direction will be decided upon by the optimization problem. The treatment here follows that of Ref. 12 in which more details can be found. The loiter pattern is described by equally spaced sample points $\boldsymbol{p}_{R,\theta}$ along the circle. By introducing a rotation matrix $\mathbf{R}(\theta)$, these points can be expressed as linear transformations of the last state $\boldsymbol{x}_T = [\boldsymbol{p}_T'\ \boldsymbol{v}_T']'$ in the planning horizon, which is the ingress state of the loiter. For the right circle $\mathcal{C}_R(\boldsymbol{x}_T)$, we obtain:

$$\begin{aligned} \boldsymbol{p}_{R,\theta} &= \boldsymbol{p}_R + \mathbf{R}(\theta)(\alpha_c \boldsymbol{v}_T^{\perp}) \\ &= (\boldsymbol{p}_T - \alpha_c \boldsymbol{v}_T^{\perp}) + \mathbf{R}(\theta)(\alpha_c \boldsymbol{v}_T^{\perp}) \\ &= \boldsymbol{p}_T + \alpha_c(\mathbf{R}(\theta) - \mathbf{I})\boldsymbol{v}_T^{\perp} \end{aligned} \tag{13}$$

in which $\boldsymbol{p}_R$ is the center of the circle. The parameter α_c is a scaling factor that is determined by the maximum turn rate of the aircraft. We thus let the radius of the loiter circle scale linearly with the magnitude of the

ingress velocity $\boldsymbol{v}_T$. As such, by slowing down or speeding up, the aircraft has the flexibility to adapt the size of its loiter circle to the state of the environment, i.e. to the trajectories and locations of the loiter patterns of the other aircraft. Note, however, that the resulting turn radius corresponding to a particular ingress speed is overestimated, thus introducing some conservatism in the loiter dynamics.

Similarly, any point $\boldsymbol{p}_{L,\theta}$ along the left loiter circle $\mathcal{C}_L(\boldsymbol{x}_T)$ can be expressed as:

$$\begin{aligned} \boldsymbol{p}_{L,\theta} &= \boldsymbol{p}_L - \mathbf{R}(\theta)(\alpha_c \boldsymbol{v}_T^{\perp}) \\ &= (\boldsymbol{p}_T + \alpha_c \boldsymbol{v}_T^{\perp}) - \mathbf{R}(\theta)(\alpha_c \boldsymbol{v}_T^{\perp}) \\ &= \boldsymbol{p}_T - \alpha_c(\mathbf{R}(\theta) - \mathbf{I})\boldsymbol{v}_T^{\perp} \end{aligned} \tag{14}$$

with $\boldsymbol{p}_L$ again the center of the circle. In the following, these geometric relations will be used to express the avoidance constraints.

C. Avoidance Constraints

We assume here for simplicity that the conflict zone $\mathcal{R}_i$ of all aircraft i is a circle of fixed radius r_{reach}, where r_{reach} is the maximum distance that can be traveled over the length of the planning horizon, including a margin for the loiter circle. Denoting the radius of the loiter circle when flying at v_{max} by r_{max}, a simple calculation yields $r_{reach} = \sqrt{(T\Delta t v_{max} + r_{max})^2 + 4r_{max}^2}$. Note that this is the most conservative assumption: in general, an aircraft might not be able to reach the full area covered by the circle, and the shape of the conflict zone will typically depend on the initial velocity $\boldsymbol{v}_{i,0}$.

The avoidance set $\mathcal{J}_i$ can then easily be determined: it consists of all aircraft that are within a distance of $2r_{reach}$ of aircraft i at time step t. In addition, Assumption 1 reduces to all aircraft being separated by a distance greater than $2r_{reach}$: $\forall i, j \in \{1 \dots K\}, i \neq j : \|\boldsymbol{p}_{i,0} - \boldsymbol{p}_{j,0}\| > 2r_{reach}$. As discussed in Steps 2a and 5 of the trajectory planning and conflict resolution algorithm, when the conflict set $\mathcal{S} \neq \emptyset$, the aircraft will communicate their plans $\mathcal{P}_i$ to one another. Since we assumed that all aircraft can accurately follow the planned trajectories, their waypoints and loiter circles can be considered as translating obstacles with known motion by the aircraft that is currently planning. We now specify these constraints in more detail.

1. *Regular Trajectory Points*

Each trajectory point $\boldsymbol{p}_{j,k}$ of aircraft $j \in \mathcal{J}_i$ is considered an obstacle that is present at time step k in the planning horizon, if that point lies in the conflict zone $\mathcal{R}_i$ of the planning aircraft i, i.e. when $\boldsymbol{p}_{j,k} \in \mathcal{R}_i$. Denote the set of time steps for which the latter, holds as $\mathcal{T}_j \subseteq \mathcal{T} = [0, \dots, T]$. Due to the discrete time nature of the trajectories, the waypoints are considered square obstacles of dimension $2d_s = 2(\max(v_{max}\Delta t, d_{safe}) + v_{max}\Delta t)$, where d_{safe} is the required safety distance around each aircraft. The lower left corner of the waypoint obstacle $\boldsymbol{p}_{j,k} = (x_{jk}, y_{jk})$ is then given by $(x_{min,jk}, y_{min,jk}) = (x_{jk} - d_s, y_{jk} - d_s)$, the upper right corner by $(x_{max,jk}, y_{max,jk}) = (x_{jk} + d_s, y_{jk} + d_s)$.

For the trajectory points (x_{ik}, y_{ik}), $k \in \mathcal{T}_j$ – that can include points on the loiter pattern if the aircraft is using its backup plan,– the required safety distance with all other aircraft is then guaranteed at all times during the planning horizon, if these points satisfy the following set of constraints:[7]

$$\begin{aligned} \forall j \in \mathcal{J}_i,\ \forall k \in \mathcal{T}_j : & \\ x_{ik} &\leq x_{min,jk} + Mb_{jk1} \\ -x_{ik} &\leq -x_{max,jk} + Mb_{jk2} \\ y_{ik} &\leq y_{min,jk} + Mb_{jk3} \\ -y_{ik} &\leq -y_{max,jk} + Mb_{jk4} \\ \sum_{n=1}^{4} b_{jkn} &\leq 3 \\ b_{jkn} &\in \{0, 1\} \end{aligned} \tag{15}$$

Here, b_{jkn} are binary variables and M is a sufficiently large positive number. The last constraint ensures that at least one of the inequality constraints is active, thereby guaranteeing that the trajectory point (x_{ik}, y_{ik}) lies outside the waypoint obstacles of the other aircraft. Note that the avoidance constraints presented here are *hard* constraints, in contrast to potential function methods where safety cannot be guaranteed.

2. Loiter Points

To ensure that the loiter circle of the planning aircraft i does not intersect with that of any of the aircraft $j \in \mathcal{J}_i$, we derive equivalent constraints for the sample points along the circle. Assume that there are $N_L = \lceil \frac{2\pi}{\theta_s} \rceil$ of these, where θ_s denotes the discretization angle. Let the loiter circle of each aircraft $j \in \mathcal{J}_i$ be contained within a square with lower left corner $(x_{min,jL}, y_{min,jL})$ and upper right corner $(x_{max,jL}, y_{max,jL})$. These dimensions include a safety boundary that accounts for the continuous segments of the loiter circle of aircraft i between its discrete sample points (see Ref. 12 for details). The two corner points, together with the time step at which the loiter is initiated, is the only information that needs to be exchanged about the loiter pattern.

The loiter square that needs to be avoided by the loiter points of the planning aircraft i can now be considered a static obstacle. By introducing a binary variable d that selects either the right or left circle, and by substituting expressions (13-14) for the sample points in the avoidance constraints (15), we obtain:

$\forall j \in \mathcal{J}_i, \forall l \in [1, \ldots, N_L]$:

$$\begin{cases} x_{iT} - \alpha_c (\cos l\theta_s - 1)\, \dot{y}_{iT} - \alpha_c (\sin l\theta_s)\, \dot{x}_{iT} & \leq & x_{min,jL} + Mb_{jl1} + Md \\ -x_{iT} + \alpha_c (\cos l\theta_s - 1)\, \dot{y}_{iT} + \alpha_c (\sin l\theta_s)\, \dot{x}_{iT} & \leq & -x_{max,jL} + Mb_{jl2} + Md \\ y_{iT} - \alpha_c (\sin l\theta_s)\, \dot{y}_{iT} + \alpha_c (\cos l\theta_s - 1)\, \dot{x}_{iT} & \leq & y_{min,jL} + Mb_{jl3} + Md \\ -y_{iT} + \alpha_c (\sin l\theta_s)\, \dot{y}_{iT} - \alpha_c (\cos l\theta_s - 1)\, \dot{x}_{iT} & \leq & -y_{max,jL} + Mb_{jl4} + Md \end{cases} \tag{16}$$

$$\begin{cases} x_{iT} + \alpha_c (\cos l\theta_s - 1)\, \dot{y}_{iT} + \alpha_c (\sin l\theta_s)\, \dot{x}_{iT} & \leq & x_{min,jL} + Mb_{jl1} + M(1-d) \\ -x_{iT} - \alpha_c (\cos l\theta_s - 1)\, \dot{y}_{iT} - \alpha_c (\sin l\theta_s)\, \dot{x}_{iT} & \leq & -x_{max,jL} + Mb_{jl2} + M(1-d) \\ y_{iT} + \alpha_c (\sin l\theta_s)\, \dot{y}_{iT} - \alpha_c (\cos l\theta_s - 1)\, \dot{x}_{iT} & \leq & y_{min,jL} + Mb_{jl3} + M(1-d) \\ -y_{iT} - \alpha_c (\sin l\theta_s)\, \dot{y}_{iT} + \alpha_c (\cos l\theta_s - 1)\, \dot{x}_{iT} & \leq & -y_{max,jL} + Mb_{jl4} + M(1-d) \end{cases} \tag{17}$$

$$\begin{aligned} \sum_{n=1}^{4} b_{jln} &\leq 3 \\ b_{jln},\ d &\in \{0,1\} \end{aligned} \tag{18}$$

Here, the index l denotes the angle around the circle, and does not necessarily correspond to the exact position of the aircraft on the circle after l time steps. Satisfying the above constraints ensures that no two loiter circles intersect; together with constraints (15) and the backup plan principle, they guarantee safety of each aircraft at all times.

D. Cost Function

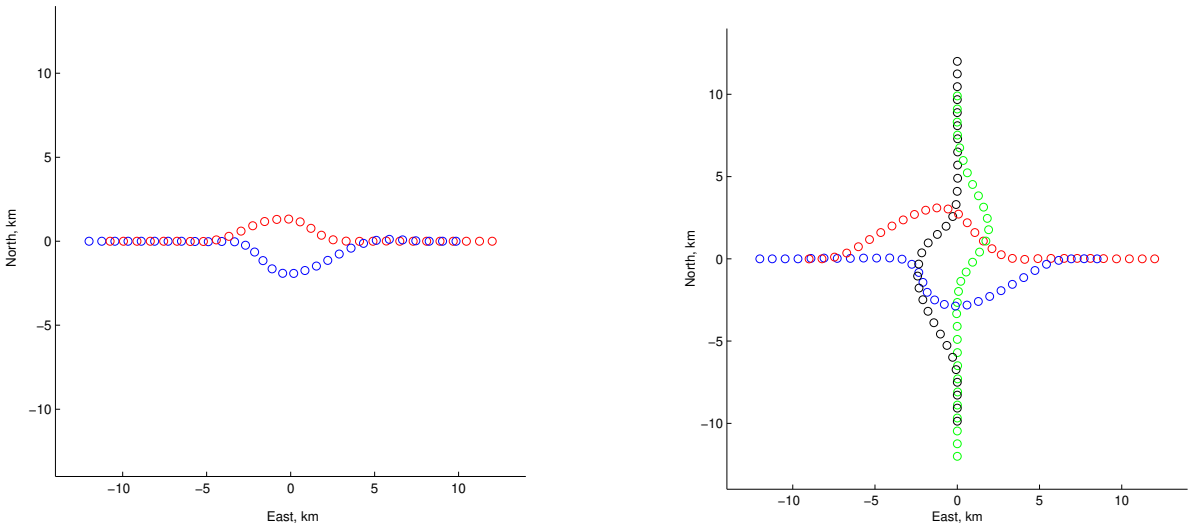
Combined with an appropriate (piecewise) linear cost function, the state space equations (6) together with the dynamic and kinematic constraints (8-12), the avoidance and loiter constraints (15-18), constitute a MILP to be solved online at each iteration of the safe trajectory planning algorithm as outlined in Section III. In our simulations, we used the following cost function for each aircraft separately:

$$\min_{\boldsymbol{x}_k, \boldsymbol{u}_k} J_T = \sum_{k=1}^{T} \left(\boldsymbol{q}' |\boldsymbol{x}_k - \boldsymbol{x}_f| \right) - r(\boldsymbol{p}_f - \boldsymbol{p}_0)' \boldsymbol{v}_k \tag{19}$$

It aims at proceeding towards the destination by *i*) minimizing the 1-norm of the difference with the desired state $\boldsymbol{x}_f$, and *ii*) maximizing the scalar product of the velocity vector with the vector $(\boldsymbol{p}_f - \boldsymbol{p}_0)$, indicating the direction from the initial position $\boldsymbol{p}_0$ to the destination $\boldsymbol{p}_f$. Since the latter remains constant over the planning horizon, the scalar product is linear. Its effect is twofold: the vehicle will redirect itself to aim straight for the goal while flying as fast as possible. As such, this cost function mimics a minimum time solution. Finally, $\boldsymbol{q}$ and r are weights that can be tuned appropriately.

V. Results

We now present some example scenarios to which we applied the proposed trajectory planning algorithm. The parameters of the aircraft used in our simulations were the following: $v_{max} = 160$ m/s, $v_{min} = 130$ m/s, $\tau = 5$ s, $k = 5$, $a_{fwd,max} = 15$ m/s^2 and $a_{lat,max} = 13.96$ m/s^2. The latter corresponds to a maximum turn



(a) Safe trajectories of 2 aircraft.

(b) Safe trajectories of 4 aircraft.

Figure 2. Simulation results for decentralized conflict resolution of 2 and 4 aircraft for 30 time steps of 5 s each.

rate of 5 deg/s. We simulated all trajectories for 30 iterations of the algorithm with a planning horizon of $T = 5$ time steps of $\Delta t = 5$ s each. The number of loiter sample points was set to $N_L = 8$, the number of linear inequalities to approximate circular constraints to $N = 16$. For the avoidance constraints, we used a required safety distance of $d_{safe} = 1.5$ km for each aircraft, resulting in square waypoint obstacles of size $2d_s = 4.6$ km. The simulation results were obtained on a Pentium 4 with 2.2 GHz clock speed, using the CPLEX 9.0 MILP solver[20] with an AMPL[21] and Matlab interface.

In the first scenario (see Figure 2a), two aircraft are flying in opposite directions and are bound to encounter each other at the origin. Aircraft 1 starts in position $(-12 \text{ km}, 0 \text{ km})$ flying West at 150 m/s, and is headed for a waypoint at $(12 \text{ km}, 0 \text{ km})$. Aircraft 2 does the opposite and starts at $(12 \text{ km}, 0 \text{ km})$, flying East at 150 m/s. When they encounter each other in the middle, they safely resolve the conflict by both turning right. Figure 3 shows the computation times of both aircraft at each iteration: with a maximum of 0.77 s, they are clearly within the timing constraints of the algorithm ($t_f - t_s < \Delta t/2 = 2.5$ s).

The second scenario (see Figure 2b) involves 4 aircraft. Aircraft 3 and 4 are now flying South and North at 150 m/s, starting in $(0 \text{ km}, 12 \text{ km})$ and $(0 \text{ km}, -12 \text{ km})$ respectively. Again, the conflict in the middle is safely resolved within the timing constraints of the algorithm (see Figure 4). Theoretically, each aircraft now has at most $\Delta t/4 = 1.25$ s available, but using a CPLEX computation time limit of 1.0 s, an acceptable solution was always found within 1.1 s. Finally, Figure 5 displays the (non-intersecting) loiter boxes of the 4 aircraft for the first 9 time steps (corresponding to 45 s).

VI. Conclusion and Future Work

We presented a framework based on Mixed Integer Linear Programming for provably safe, decentralized trajectory planning of multiple (autonomous) aircraft. A receding horizon planning strategy was adopted for each aircraft individually, that accounts for the trajectories of the neighboring aircraft. Safety at all time steps was guaranteed by constraining the intermediate plans of all aircraft to terminate in loiter circles that do not intersect. Conflicts between multiple aircraft were resolved in a decentralized fashion while maintaining an *a priori* safe backup plan for each aircraft. The detailed MILP formulation was given and simulation scenarios with timing results were presented.

Future work includes extending the framework to account for robustness against uncertainties in the trajectory tracking performance of the aircraft, including static no-fly zones in the simulations, considering heterogeneous sets of UAVs with highly agile vehicles such as miniature helicopters, and testing the algorithm

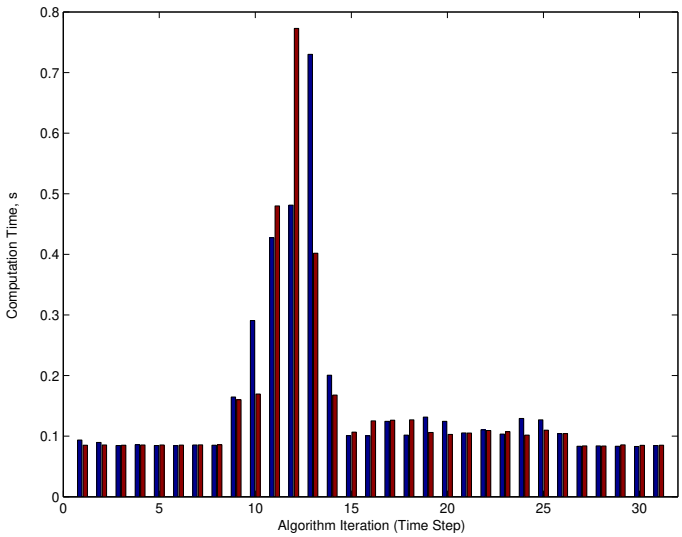


Figure 3. Computation times at each iteration for the 2 aircraft scenario of Figure 2a.

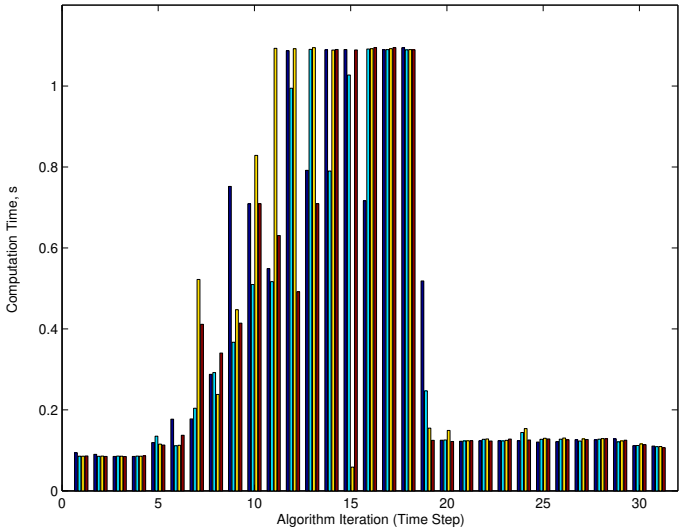


Figure 4. Computation times at each iteration for the 4 aircraft scenario of Figure 2b.

on MIT's multiple UAV test-bed. In addition, a performance comparison with centralized methods will be made.

Acknowledgments

This research was funded under the DARPA Software Enabled Control Program, AFOSR contract #F33615-01-C-1850, and the DARPA Complex Adaptive Networks For Cooperative Control Program, AFOSR contract #009628-001-03-132.

References

[1]Tomlin, C., Pappas, G., and Sastry, S., "Conflict Resolution for Air Traffic Management: a Study in Multi-Agent Hybrid Systems," *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, 1998, pp. 509–521.

[2]Tomlin, C., Mitchell, I., and Ghosh, R., "Safety Verification of Conflict Resolution Maneuvers," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 2, No. 2, 2001.

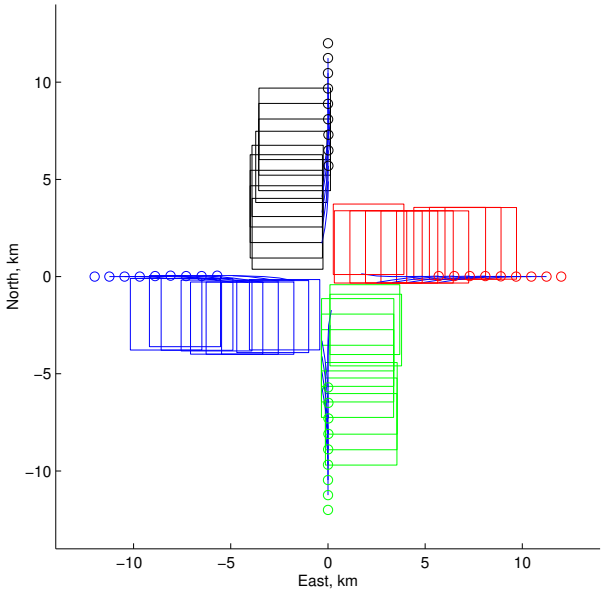


Figure 5. Safe trajectories and corresponding loiter boxes for the 4 aircraft scenario during the first 9 time steps (corresponding to 45 s).

[3]Mitchell, I., Bayen, A., and Tomlin, C., "Computing Reachable Sets for Continuous Dynamic Games using Level Set Methods," *Submitted to the IEEE Transactions on Automatic Control*, Jan. 2003.

[4]Bayen, A., Cruck, E., and Tomlin, C., "Guaranteed Overapproximations of Unsafe Sets for Continuous and Hybrid Systems: Solving the Hamilton-Jacobi Equation Using Viability Techniques," *Hybrid Systems: Computation and Control*, edited by C. Tomlin and M. Greenstreet, Vol. 2289 of *Lecture Notes in Computer Science (LNCS)*, 2002, pp. 90–104, Springer Verlag.

[5]Frazzoli, E., Mao, Z.-H., Oh, J.-H., and Feron, E., "Resolution of Conflicts Involving Many Aircraft via Semidefinite Programming," *AIAA Journal of Guidance, Control and Dynamics*, Vol. 24, No. 1, 2001, pp. 79–86.

[6]Raghunathan, A., Gopal, V., Subramanian, D., Biegler, L., and Samad, T., "Dynamic Optimization Strategies for 3D Conflict Resolution of Multiple Aircraft," *Submitted to AIAA Journal of Guidance, Control and Dynamics*, 2003.

[7]Schouwenaars, T., DeMoor, B., Feron, E., and How, J., "Mixed Integer Programming for Multi-Vehicle Path Planning," *Proc. European Control Conference (ECC'01)*, Porto, Portugal, Sept. 2001.

[8]Richards, A. and How, J., "Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming," *Proc. 2002 American Control Conference*, Anchorage, AK, May 2002.

[9]Pallottino, L., Feron, E., and Bicchi, A., "Conflict Resolution Problems for Air Traffic Management Systems Solved with Mixed Integer Programming," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 3, No. 1, 2002, pp. 3–11.

[10]Oishi, M., Tomlin, C., Gopal, V., and Godbole, D., "Addressing Multiobjective Control: Safety and Performance through Constrained Optimization," *Hybrid Systems: Computation and Control*, edited by M. D. Benedetto and L. . A. Sangiovanni-Vincentelli, Vol. 2034 of *Lecture Notes in Computer Science (LNCS)*, Springer Verlag, 2001, pp. 459–472.

[11]Hu, J., Prandini, M., and Sastry, S., "Optimal Coordinated Maneuvers for Three-Dimensional Aircraft Conflict Resolution," *AIAA Journal of Guidance, Control and Dynamics*, Vol. 25, No. 5, 2002, pp. 888–900.

[12]Schouwenaars, T., How, J., and Feron, E., "Receding Horizon Path Planning with Implicit Safety Guarantees," *Proc. 2004 American Control Conference*, Boston, MA, June 2004.

[13]Schouwenaars, T., *Mixed Integer Programming for Optimal Collision-Free Path Planning of Autonomous Vehicles*, Master's thesis, Katholieke Universiteit Leuven, Department of Electrical Engineering, Leuven, Belgium, May 2001.

[14]D.H. Shim, H. K. and Sastry, S., "Decentralized Nonlinear Model Predictive Control of Multiple Flying Robots in Dynamic Environments," *Proc. 44th IEEE Conference on Decision and Control (CDC'03)*, Maui, HI, Dec. 2003.

[15]Inalhan, G., Stipanovic, D., and Tomlin, C., "Decentralized Optimization, with Application to Multiple Aircraft Coordination," *Proc. 41st IEEE Conference on Decision and Control*, Las Vegas, NV, Dec. 2002.

[16]Bellingham, J., Richards, A., and How, J., "Receding Horizon Control of Autonomous Aerial Vehicles," *Proc. 2002 American Control Conference*, Anchorage, AK, May 2002.

[17]Royer, E. and Toh, C.-K., "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," *IEEE Personal Communications*, Vol. 6, No. 2, 1999, pp. 46–55.

[18]Toh, C.-K., *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall Publishers, 2001.

[19]Schouwenaars, T., Mettler, B., Feron, E., and How, J., "Hybrid Architecture for Full-Envelope Autonomous Rotorcraft Guidance," *59th Annual Forum of the American Helicopter Society*, Phoenix, AZ, May 2003.

[20]*ILOG CPLEX 9.0 User's guide*, ILOG, 2003.

[21]Fourer, R., Gay, D. M., and Kernighar, B. W., *AMPL, A modeling language for mathematical programming*, The Scientific Press, 1993.