Convergence of Peer-to-Peer Collision Avoidance among Unmanned Aerial Vehicles

Přemysl Volf, David Šišlák, Michal Pěchouček and Magdalena Prokopová
Gerstner Laboratory, Department of Cybernetics
Czech Technical University in Prague
Technická 2, Prague 6, 166 27 Czech Republic

E-mail: {volf,pechouc,prokopova}@labe.felk.cvut.cz, sislakd@fel.cvut.cz

Abstract

In this article we study the theoretical aspects of the collision avoidance among the collectives of unmanned aerial vehicles (UAVs) engaged in the free flight operation. The free flight based operations do not provide collision-free trajectories as a flight specification to the UAVs. Collision avoidance of the trajectories is implemented by means of peer-to-peer negotiations among the individual UAVs. In this paper we study the role of interaction in the collision avoidance process. We prove theoretically convergence of the specific negotiating protocol that has been deployed in a practical implementation of the software prototype of the distributed collision avoidance system.

1 Introduction

Many UAV relief operations require the See & Avoid capability as specified in [1]. When distributed among multiple autonomous aircrafts, the See & Avoid capability allows to realize the benefits of the free flight concept – an approach for autonomous routing of aricrafts based on local collision avoidance mechanisms. In free-flight operations the flight trajectory of each individual aircraft is planned without the awareness of the future operations of other vehicles.There are two main motivations behind the free flight approach: (1) data on trajectories of all airplanes are not available centrally, and/or (2) in highly dynamic environments, flight trajectories need to be replanned frequently. Upon collision detection an aircraft engages in a negotiation process. Such an approach allows efficient operation of dynamically tasked unmanned aerial vehicles (UAV). While this approach is particularly suited for the real-time control of vehicles, this concept can also be used for simulation purposes.

In this article we analyze the performance of the developed, multi-agent collision avoidance architecture. We provide results of the theoretical analysis of the convergence of the negotiation process among multiple autonomous agents controlling the individual UAVs.

First, we explain the problem of collision avoidance. Later, we explain the convergence problem. *The collision avoidance problem:* Each UAV is controlled by a single, dedicated autonomous agent. The collision avoidance problem is to design such an algorithm that provides optimal (e.g. minimal trajectory increment) collision avoidance manoeuvres in the environment while respecting constrains on the velocity, altitude or admissible range of changes of direction of individual flying vehicles. Simulation is performed within a flexible multi-layer collision avoidance architecture which supports the integration of several different deconfliction algorithms described in section 2. Properties of this architecture were verified by several scalability experiments [9].

The Convergence problem: In practical deployments of competitive, multi-agent algorithms we often face the problem of communication cycles. An example is a *monotonic concession protocol* (MCP) [10], where the parties are incrementally refining their proposals towards a joint solution. In this article we prove that algorithms we designed cannot cycle infinitely and that UAVs will find a collision avoidance manoeuver in a finite time.

Our goal is to build a formal model of the distributed agent-based algorithm that we have already developed and to formally prove its convergence with as few restrictions and limitations as possible, and to describe its correspondence with algorithm implemented in our framework. We formalize the utility-based collision avoidance algorithm and select scenario where group of airplanes is forced to fly through a narrow tunnel. At the beginning they have the same speed and distance to tunnel starting point and thus all should collide if no collision avoidance algorithm is used. Aircrafts are restricted to apply only speed up and slow down manoeuvres. We show convergence of distributed

0-7695-3027-3/07 $25.00 © 2007 IEEE
DOI 10.1109/IAT.2007.75

pairwise utility-based algorithm under all circumstances.

Cooperative collision avoidance is topic of many research projects taking different approaches. Nevertheless only limited number of researchers try to analyze the collision avoidance theoretically, while the majority contribute to empirical evaluation of the specific algorithms. The research community also aims at comparing each approach using different collision metrics [6] and negotiation protocols. The most important metric is number of collisions remaining after the algorithm is performed. The problem solved in this paper is whether the algorithm can guarantee desired behavior at any case under given assumptions.

Krozel compares three different approaches to conflict resolution – one centralized and two distributed methods [6]. Centralized strategy emphasizes stability of the system rather than efficiency. Decentralized algorithms use myopic and look-ahead strategies. First algorithm prefers efficiency to stability. Second also prefers efficiency, but takes into consideration stability as well. Hill *et al.* uses satisficing approach using game theory [3]. The method is based on dual utility - *selectability* to characterize effectiveness and *rejectability* to characterize inefficiency. Algorithm is tested on different scenarios with high number of airplanes. Two flows of air traffic that intersect at a fixed point represent a problem analyzed in [7][2]. Problem is solved by single heading change at the beginning. The results show that only small manoeuvres are needed to avoid collision. Kosecka, Tomlin *et al.* use potential and vortex fields to solve collision up to four aircrafts [5]. Generalized overtake and head-on manoeuvres may solve all two-aircrafts collisions. For more airplanes they propose manoeuvre called roundabout. Holdsworth [4] models an escape trajectories generated by different algorithms using Brisbane Model. It uses tesselated space with aircraft's position represented as one tile. This model with probability and risk calculation is used to identify cases where algorithms fail for further analysis.

We can divide these works into three groups. First group tries to simulate or model developed algorithm in different scenarios with a lot of airplanes and the results are usually extensive measurements with analysis of properties. Second group analyzes small number of airplanes (usually up to four) and formally proves collision avoidance ability of suggested algorithm at any situation, usually using geometry and/or restricted manoeuvres. Third group set up some specific scenario and proves correctness of algorithm for this case, usually applies single manoeuvre for each airplane. We try to combine all three approaches. We have framework matching first group and we will prove selected algorithm as in third group.

2 ATC Collision Avoidance Architecture

We have developed and implemented complex simulation framework of the ATC (Air Traffic Control) system built on top of agent-based platform $\mathcal{A}$-**globe** [8].

We have built a *multi-layer collision avoidance module* as a part of the pilot agent [9]. This module[1] allows to use and combine several algorithms used for collision avoidance. Collision avoidance module can be configured to use several algorithms together. These algorithms can be either cooperative or noncooperative. Actual selection depends on the configuration, type of other airplane, trust, time to collision etc.

The noncooperative algorithms do not assume any collaboration or information from other pilot agents. The cooperative algorithms suppose pilots are willing to cooperate to solve collisions. This can be further divided into self-interested (optimize its own goals) or altruistic (optimize total social welfare among group of airplanes).

2.1 Utility-Based Algorithm

The presented formal model is based on the utility-based collision avoidance algorithm. This negotiation protocol solves collisions pairwise. The utility function is used for differentiating several different deconfliction maneuvers from the point of view of the individual aircraft (e.g. fuel consumption, delay at mission points, risk factors, etc).

Algorithm starts with selecting the soonest conflict. It generates a set of possible flight trajectories for each airplane using predefined manoeuvres (e.g. turn left, turn right, turn up, turn down, speed up, slow down). This manoeuvres are constructed to avoid the conflict. Each generated flight trajectory is tested for conflict with all other airplanes. If a collision with any airplane is found that would happen sooner than the currently solved one, such trajectory is removed from the generated set. Afterwards, sets for both airplanes are combined to create all possible pairs of trajectories (cartesian product of these sets). Each pair is checked whether mutual collision persists. If there are some valid pairs, the best one is selected. Such selection can be based for example on sum of the utility value of each trajectory to reach maximum social welfare. If there is not any valid pair, the manoeuvres are applied again to generate wider range of trajectories. Newly generated and checked flight plans are added to the previously generated set. This process is repeated until a valid pair of generated trajectories is found and applied.

When single collision is solved, the next soonest collision is selected. Iterations continue until all collisions are solved.

[1] The architecture is domain independent.Its modular architecture with defined interfaces can be adopted to deployment also in the other domains with autonomous entities. There is only one strong prerequisite that the entity must know detailed information about its future motion plans including time information.

3 Deployment Scenario

We have been testing the operation of the ATC Collision Avoidance Architecture on several high density scenarios, such as random high traffic in a limited area, fixed operation planned so that multiple collisions threats of high number of UAVs are involved, etc.). We have decided to base our formal model on the *tunnel scenario*, as an example of situation where deconfliction problem is limited by space constrains[2]. The scenario contains n airplanes willing to fly from their starting positions through the tunnel to their destinations, see Figure 1. All airplanes fly at same altitude and they cannot manoeuvre and avoid collisions by changing this altitude. The tunnel is so narrow, that no more than one airplane can fly through it at a time.

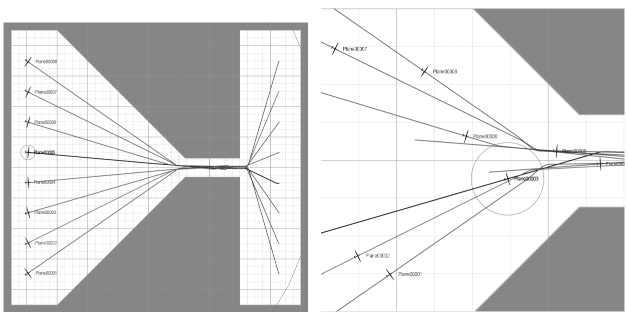

Figure 1. Tunnel scenario. Airplanes fly through tunnel. Circle around airplane means safety zone.

By closer look at the scenario, we can see that to avoid collisions in the tunnel all airplanes have to fly at the same speed to the tunnel starting point and than just keep their speeds to fly through it. Collisions will be avoided by careful arranging aircraft's time of arrival to the tunnel and keeping minimal distance from other aircrafts equal to safety zone size. Without loss of generality, we suppose only speed ups and slow downs are allowed. Any other manoeuvre that would change airplane trajectory would result in different time of arrival to the tunnel and therefore it wouldn't bring any additional possibility how to avoid collision.

We expect worse case scenario where airplanes are set to such starting positions, that if we sort them by times of arrival to the tunnel, time interval between any two successive airplanes is not longer than time needed to fly over its safety zone, see Figure 2.

This case can be also used as landing scenario, where several airplanes want to land at the same airport with only one runway. Apparently only one airplane at a time can line up with the runway and land, therefore airplanes try to adjust the time of their arrivals to the airport.

[2] `http://agents.felk.cvut.cz/atg-videos/theoretical/`

3.1 Assumptions and Objectives of the Model

We have built formal model of the tunnel scenario. This model should keep properties and complexity of the simulated scenario, but it should also provide us with the possibility to use formal methods to prove convergence.

It is sufficient to model only the first phase of the flight from starting positions to the tunnel starting point. The only important aspect is the time of arrival of each airplane to the tunnel, thus all flights can be modeled as a intervals on the time axis representing airplane's safety zone, see Figure 2. As a simplification we regard starting position of each airplane only as time needed to fly to the tunnel using its original speed. Note that we lose information about position of the airplanes. We do not know when and with the first collision occur in this model, so the closest neighbor at the time axis does not need to be airplane with the soonest collision. Only information is whether or not collision will occur.

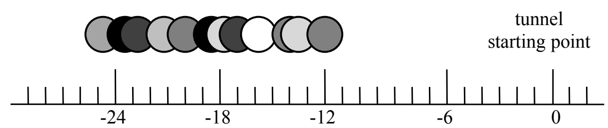

Figure 2. Tunnel scenario modeled as time axis. Airplanes are on their starting positions. Overlapping airplanes will have a collision in the starting point.

We assume that at the beginning speed of all airplanes is identical and they will enter the tunnel at this speed. Speed up and slow down manoeuvres are transformed to adjustments of aircraft's time of arrival to the tunnel (time jumps). Thus speed up manoeuvre does not mean, that airplane will increase its speed, but that it will arrive to the tunnel sooner maintaining its original speed. This can be done by speeding up for necessary time to gain time and then slow down to the original speed. Slow down is defined in similar way. Denote *shift forward* manoeuvre, when the flight time to the tunnel is decreased and *shift backward* manoeuvre when the flight time to the tunnel is increased. We assume number of the shifts in either way is not limited. Size of this zone is defined as the time needed to fly through the safety zone of an aircraft at the original speed.

Utility function is defined as local optimization, this means that utility value is optimized for each deconfliction separately. Every airplane remembers its actual position at the beginning of each deconfliction and this position generates the best value of the function. Utility value is getting worse with (time) distance from actual position in the same way in both directions. Thus every airplane tries to minimize necessary change from its actual position. If several minimal choices are possible, random solution is selected.

It is guaranteed that airplanes trust each other and offer true and reliable information. We also assume that each airplane has access to the information about flight plans of all other airplanes. We assume communication is parallel, but synchronous. Thus only negotiation of one pair can be performed at a time. This prevents multiple changes of the flight plans and possible reverting some of them. We assume communication is reliable.

4 Formal Proof of Convergence

Definition 1 (First (last) airplane) *First (last) airplane is airplane with the earliest (latest) time of arrival to the tunnel starting point. If there are several airplanes with the earliest (latest) time, all of them are considered as the first (last) airplane.*

Definition 2 (State of the problem) State *of the deconfliction problem is defined by positions of all airplanes at the time axis.*

Definition 3 (Step) *One* Step *of the algorithm is application of utility based negotiation to pair of aircrafts and then performing appropriate changes (shifts) to their positions. One step transforms one state to another.*

Definition 4 (Cycle) *Define* cycle *as a sequence of the steps, where the state of the deconfliction problem is the same before and after performing these steps.*

Definition 5 (Restricting neighbors) *All airplanes are tagged by either restricting or non-restricting tag for each deconfliction. Suppose an airplane A is trying to shift from its original position to the new position p (to solve the collision c). Airplane X is marked as restricting for the airplane A, collision c (its time) and the new position p, if airplane A can not solve collision c by moving to the position p because collision with X would arise sooner then c. Otherwise (if there is no collision taking place sooner than collision c, including no collision at all) the airplane X is marked as non-restricting. Desired shift to the new position is allowed only if no restricting airplane exist for this position.*

Definition 6 (Constants and variables) *Constants (C) are same all the time, variables (V) can vary. Denote*

- sz *(C) as size of safety zone measured in time units.*
- d *(V) as (time) distance between airplanes on the time axis, ie. difference in arrival times of airplanes to the tunnel.*
- D *(V) as (time) distance between the first and the last airplane.*
- ms *(C) as manoeuvre step, minimal size of the shift forward or backward. Airplanes are shifting only in multiplies of this value.*
- $ts(D) = \frac{D}{ms}$ *(V) as a total number of positions between the arrival to the tunnel of the first and the last airplane.*
- $s(n, D)$ *(V) as number of all possible states between actual first and last airplane.* $s(n, D) = ts(D)^n$

Lemma 1 *The position of the first (last) airplane cannot be shifted backward (forward).*

Proof. Airplane shifts forward or backward in time only as a result of a deconfliction. Conflict for the *first* airplane can arise only with airplane with later time of arrival (from the definition). Suppose that the *first* airplane will shift backward. Figure 3 shows all possible changes of the airplanes[3].

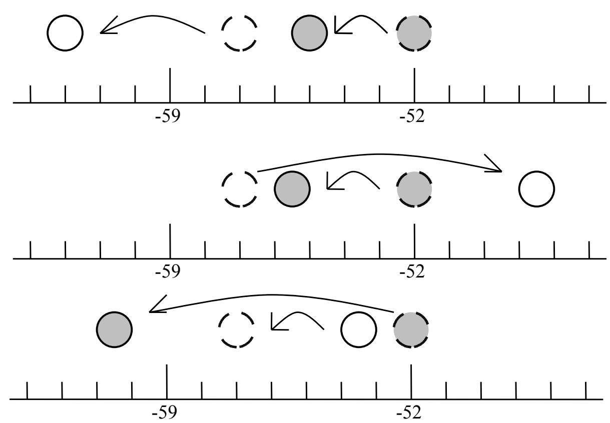

Figure 3. Deconfliction of the *first* airplane (grey). Original positions are with dashed edges.

In the first case, there is no reason for the *first* airplane (grey) to shift backward, because the new utility value will be worse than utility value for its original place.

In the second case, airplanes switch their positions and second (white) airplane gets in front of the grey one. Thus the white airplane is marked as new *first* airplane and thus position of the *first* airplane is shifted forward.

In the third case, airplanes are switched, but white is still behind original position of the *first* airplane. More optimal change for the *first* airplane would be to shift forward instead of backward, because it leads to smaller change from its original position.

Also situation where two airplanes (both *first*) are solving their collision can appear. Although even in this case there is no reason to shift both airplanes backward. This situation is analogous to the first case. □

Lemma 2 *Assume each airplane solves its soonest collision first, then cycle cannot arise.*

[3] size of the safety zone is defined by larger lines at the time axis. It has size of 7 in this Figure.

Proof. We prove this lemma by contradiction. Assume that cycle exists. This cycle cannot contain any collision free state, because algorithm would immediately stop and cycle wouldn't be formed. Therefore all states in the cycle contain a collision.

Each step of algorithm is a result of solving some collision. Denote c as a first collision being solved in the first state of the cycle. Consequently after solving collision c and moving to the second state collision c doesn't exist any more. No collision can arise sooner or in the same time as a result of a deconfliction (as a property of the utility based algorithm). No state after second state can contain collision c and therefore the first and the last state is not same. □

Implication 3 *D has to be increased after $(2n-1)\cdot s(n,D)$ steps by at least ms.*

Proof. As a result of Lemma 2 after exhausting all possible states ($s(n,D)$ steps) without solving collisions either the *first* or the *last* airplane has to shift. Lemma 1 proves that the *first* airplane can shift only forward and the *last* airplane can shift only backward. Thus distance between these airplanes D has to be increased.

How long it takes to increase D for at least ms? Suppose only shifting of the *first* airplane forward after $s(n,D)$ steps. Let's mark the *first* airplane as airplane A. The position of the *first* airplane can be shifted forward in two different ways. Firstly, airplane A shifts forward for the minimal step ms, therefore the implication holds.

Secondly, other airplane (airplane B) shifts in front of the airplane A and is marked as new *first* airplane. Distance between the original position of first airplane A and the new position of first airplane B can be lower than ms and thus not sufficient. After n takeovers of the leader, at least one airplane has to be in lead twice and therefore this airplane moved at least for ms, which shows that implication holds.

Let allow also shifts of the *last* airplane backward. Position of the last airplane can change at most $n-1$ times without increasing the D by ms.

Since each shift of first/last position is performed after maximum of $s(n,D)$ and there can't be more than $n-1$ on each side without increasing the D for ms, the lemma is proved. □

Lemma 4 (Maximal separation) *As a result of the single deconfliction, the distance between any two neighboring (on the time axis) airplanes cannot be increased to value greater than*

$$d \leq 2\cdot sz + ms$$

Proof. Assume two airplanes solving their collision. Algorithm starts by each airplane generating its possible future positions. Several situation can occur.

In the first case, new positions are not restricted by any other aircraft and changes can be applied. After deconfliction the distance between airplanes will stay $d < sz + ms$.

In the second case, an aircraft restricting their movement at one side exists therefore airplanes will use space on the other side to solve the collision and the distance will stay $d < sz + ms$.

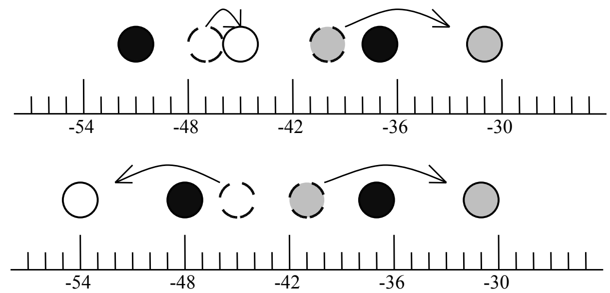

Figure 4. Figure shows deconfliction of the airplanes (grey and white), that are restricted by another (black ones). Deconflicting airplanes cannot shift their position closer than sz to any black airplane.

In the third case, there are restricting aircrafts at both sides of both airplanes, see Figure 4. Two black circles at the Figure shows restricting aircrafts closest to the pair of airplanes in collision.

Distance between restricting aircrafts is lower than $3\cdot sz$ and thus there is not enough space to solve deconfliction in between these aircrafts without arising collision with one of them. The situation is solved by extending the shift of the airplane in the conflict to reach position behind restricting aircraft(s). You can see two different cases at the Figure.

In the first situation, distance between black aircrafts is bigger than $2\cdot sz+ms$, thus one of the airplanes (grey in our case) will jump over the restricting aircraft. Now white airplane has enough space to find the best position between restricting aircrafts. It will hold minimal separation of sz between white and black airplanes and thus maximal distance between any two airplanes is not increased over $2\cdot sz+ms$.

In the second situation, no airplane from pair of airplanes in the conflict can stay between black aircrafts. But this also means that distance between restricting aircrafts is not bigger than $2\cdot sz+ms$. Otherwise there would be enough space for one airplane to stay between them. □

Implication 5 *Minimal distance D ensuring collision free state (where distance between any two airplanes is at least sz) is*

$$D_{min}(n) = (n-2)(2\cdot sz + ms) + sz$$

Optimal (shortest) distance D, to reach collision free state, is

$$D_{opt}(n) = (n-1)\cdot sz$$

Proof. Collision does not exist between any two airplanes, if distance between them is at least sz. Lemma 4 says, that distance between any two airplanes cannot be increased to distance greater than $2 \cdot sz + ms$. Thus minimal distance D, that guarantees distance between any two airplanes to be at least sz, is $D_{min}(n) \leq (n-2)(2 \cdot sz + ms) + sz$.

Distance between any two airplanes have to be at least sz. Thus minimal space to hold all airplanes without collisions is $D_{opt}(n) = (n-1)sz$. □

Theorem 6 (Convergence) *Collision free state for all airplanes is reached within finite amount of algorithm steps.*

Proof. Implication 3 shows that D has to be increased after finite amount of steps for at least nonzero distance. Implication 5 proves that $D_{min}(n)$ is sufficient to collision free state for all airplanes. Thus all collisions are solved within finite amount of the algorithm steps. □

Theorem 7 (The worst case estimate) *Collision free state is reached in maximum*

$$acs(n, D) \leq \sum_{i=D}^{D_{min}(n)} (2n-1)(\frac{i}{ms})^n$$

steps, where D is distance between the first and the last airplane in the initial state and sum step is ms.

Proof. $(2n-1) \cdot s(n, D) = (2n-1) \cdot ts(D)^n$ steps is needed at maximum to increase D at least for ms. This number of steps has to be added for every change of distance (using the step ms) between the first and last airplane from the initial D to the D_{min} when collision free state is guaranteed. □

5 Estimations and Restrictions

We proved convergence of the problem in previous section. In this section, we will make further analysis. We specify distance of the airplanes from the tunnel necessary to solve all collisions, and prove convergence under the assumption that a global optimum exists and remove constrains of constant speed.

5.1 Limitation of the Shifts.

We assume number of shifts forward or backward is unlimited though this presumption is stronger than is actually needed. The distance between the *first* and the *last* airplane is not greater than $D_{min}(n) = (n-2)(2 \cdot sz + ms) + sz$. Thus aggregated number of steps in either way is limited for every airplane and is equal to $\frac{D_{min}(n)}{ms}$.

To determine minimal distance from the starting position to the tunnel starting point for each airplane only overall maximal time shift of an airplane is necessary. This maximal time shift is equal to $D_{min}(n)$. Denote v_{orig} as original, v_{max} as maximal, v_{min} as minimal speed of the airplane. Suppose $v_{min} < v_{orig} < v_{max}$ and ignore acceleration and deceleration. Then minimal time distance from the tunnel starting point to have enough time to shift forward (speed up) is $t_{fwd}(n) = \frac{v_{max} D_{min}(n)}{v_{max} - v_{orig}}$ and to shift backward (slow down) is $t_{back}(n) = \frac{v_{min} D_{min}(n)}{v_{orig} - v_{min}}$. Therefore minimal time distance to have enough time to avoid collisions is $t_{min}(n) = \max(t_{fwd}(n), t_{back}(n))$ and corresponding space distance is $d_{min}(n) = t_{min}(n) \cdot v_{orig}$.

5.2 Using Global Optimum.

Another presumption was that the utility function was defined only as local optimization. This condition is used in lemma 1, to show desired selection of the manoeuvres. We show that utility function can also select solutions with regard to global optimal value. With the new utility function lemma 1 is not valid any more. The difference is shown at the Figure 5. In the original definition dashed green airplane (on the left) had no reason to move from its position. But with global optimum placed more at left side, green airplane will shift to the new position (solid circle).

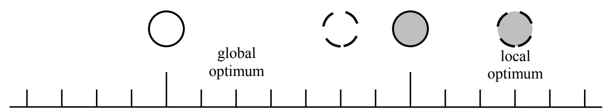

Figure 5. Figure shows different selection of the manoeuvres considering local and global optimum.

Lemma 1 is used in implication 3 although it can be proved even without it. Using lemma 2 we know, that D has to be increased or stay the same in which case the whole interval on time axis containing all airplanes have to be shifted after $s(n, D)$ steps. Shift of the whole can occur only when the *first* (*last*) airplane has its global optimum behind (in front of) its actual position. Thus actual interval has to contain global optimum of at least one airplane. Denote D_{orig} as the longest distance between any two global optimums (usually corresponding to starting positions). Size of the time interval, where all airplanes has to be at any time, is $2D_{min} + D_{orig}$ (airplanes can be shifted to the side no more than D_{min} times). As we already proved cycle cannot exist and therefore position of the interval containing all airplanes on the time axis cannot repeat. Thus number of possible positions for the interval is finite and convergence of the algorithm holds.

5.3 Different speeds.

We suppose all airplanes have the same original speed and they do not changed it. Same speed is needed to guarantee same size of (time) safety zone as a necessary condition to fly through the tunnel. What will change, if we permit different speed at the starting points, any where during the flight and disregard scenario with the tunnel and think just about flight plans with single intersection?

In this case airplanes can fly through this point with different speeds and therefore all airplanes have to know speeds of each other to determine size of a safety zone for each airplane. Algorithm used to solve a collision of a pair of airplanes remains the same, we just need to take into consideration changing safety zones. This allows us also to introduce airplanes with different sizes of the safety zone and with different minimum and maximum speeds.

We can use different airplanes to flight together and use distributed deconfliction algorithm, but then we need to redefine some variables. Minimum (maximum) speed has to be defined as the lowest (highest) speed among all airplanes. sz is defined as a minimum over all airplanes of time needed to fly with minimum speed over its safety zone. With these changes all computations mentioned above are valid.

6 Conclusions and Future Extensions

In this paper we show convergence of the distributed pairwise utility-based algorithm. We select a specific set of scenarios with strong constrains, where a group of airplanes is forced to fly through a single point in space. This represents one of the most difficult cases for collision avoidance. We assume an identical type of airplanes and that all changes of aircraft trajectories are made only by speed ups and slow downs.

We examine the properties of the described scenario and algorithm and formalize these properties into a formal model. The algorithm we are using is distributed and therefore we cannot predict the exact execution of the algorithm since agents act independently.

We provide the worst case estimate of the convergence. Finally, we relax some of the assumptions without breaking convergence of the algorithm. This model corresponds to a simulation of this scenario using the simulation framework of the ATC (Air Traffic Control) system.

It the future we are interested in several directions of our research. One of the most important aims is to make a more realistic estimate of the convergence, to determine the worst and average case, at least to prove whether the problem is polynomial or not. Another direction for the future research is to extent this proof to other scenarios with the ultimate goal to cover all possible situations that can be solved by speed changing manoeuvres. We will also try to reduce assumptions to get closer to simulations and real models.

Acknowledgment

Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-06-1-3073. The U.S. Government is authorized to reproduce and distribute reprints for Government purpose notwithstanding any copyright notation thereon. [4]. This research has been also supported by the Ministry of Education, Youth and Sports of the Czech Republic grant 6840770038 and as a part of the specific research at the CTU in Prague, under grant number CTU0706513.

References

[1] DOD. Unmanned aircraft systems roadmap 2005-2030.

[2] D. Dugail, E. Feron, and K. Bilimoria. Stability of intersecting aircraft flows using heading change maneuvers for conflict avoidance. In *Proceedings of American Control Conference*, pages 760–766, 2002.

[3] J. C. Hill, F. R. Johnson, J. K. Archibald, R. L. Frost, and W. C. Stirling. A cooperative multi-agent approach to free flight. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 1083–1090, New York, NY, USA, 2005. ACM Press.

[4] R. Holdsworth. *Autonomous In-Flight Path Planning to replace pure Collision Avoidance for Free Flight Aircraft using Automatic Dependent Surveillance Broadcast*. PhD thesis, Swinburne University, Melbourne, Australia, November 2003.

[5] J. Kosecka, C. Tomlin, G. Pappas, and S. Sastry. Generation of conflict resolution manoeuvres for air traffic management. In *Proceedings of Intelligent Robots and Systems*, volume 3, pages 1598–1603, September 1997.

[6] J. Krozel, M. Peters, K. D.Bilimoria, C. Lee, and J. S. Mitchel. System performance characteristics of centralized and decentralized air traffic separation strategies. In *4th USA/Europe Air Traffic Management R & D Seminar*, Stanta Fe, NM, December 2001.

[7] Z.-H. Mao, E. Feron, and Bilimoria. Stability and performance of intersecting aircraft flows underdecentralized conflict avoidance rules. In *Proceedings of Intelligent Transportation Systems*, volume 2, pages 101–109, 2001.

[8] D. Šišlák, M. Rehák, M. Pěchouček, M. Rollo, and D. Pavlíček. $\mathcal{A}$-**globe**: Agent development platform with inaccessibility and mobility support. In R. Unland, M. Klusch, and M. Calisti, editors, *Software Agent-Based Applications, Platforms and Development Kits*, pages 21–46, Berlin, 2005. Birkhauser Verlag.

[9] D. Šišlák, P. Volf, A. Komenda, J. Samek, and M. Pěchouček. Agent-based multi-layer collision avoidance to unmanned aerial vehicles. In J. Lawton, editor, *Proceedings of 2007 International Conference on Integration of Knowledge Intensive Multi Agent Systems*, volume KSCO 2007. IEEE, IEEE, 2007.

[10] G. Zlotkin and J. S. Rosenschein. Negotiation and task sharing among autonomous agents in cooperative domains. In N. S. Sridharan, editor, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 912–917, San Mateo, CA, 1989. Morgan Kaufmann.

[4]The views and conclusions contained herein are those of the author and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government