2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology

Multi-party Collision Avoidance among Unmanned Aerial Vehicles

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

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

Abstract

This paper addresses the problem of distributed cooperative collision avoidance that supports efficient utilization of air space shared by several autonomous unmanned aerial vehicles. The novel multi-party collision avoidance (MPCA) algorithm is described. It is compared to the iterative peer-to-peer collision avoidance (IPPCA) algorithm that iteratively optimizes social welfare. The paper provides a set of experiments and a comparison of different collision avoidance mechanisms in a multi-agent model of air traffic.

1 Introduction

The cooperative operation of a group of autonomous unmanned aerial vehicles (UAVs) fulfilling a given objective of their mission (e.g. monitoring , surveillance or more complex missions) requires the See & Avoid capability [1]. The centralized approach to the UAV collision avoidance requires permanent communication connection between aircraft and a central avoidance controller. This is a strong restriction especially in the situations when UAVs operate in a hostile environments and when they dynamically change their current tasks. In such circumstances they need to update their mission frequently and avoid collision threats. Frequent communication with the distant central controller may compromise the mission. Alternatively, the aircraft can implement collision avoidance capability in fully distributed manner. In such a case, the communication among the aircraft requires much lower intensity of the communication radio signals than in case of interacting with the centralized coordinator. Such approach is known as *free flight* concept [2] – autonomous routing of the aircrafts based on local collision avoidance mechanisms.

A number of different approaches to the decentralized air traffic separation strategies are listed in [3]. One possibility for the distributed solution is the deployment of principled negotiation [9]. There are also various approaches based on the game theory (e.g. [2]) available in the research community. Another approach is based on composition of a fully distributed solution between two aircraft based on various agent-based negotiation protocols (e.g. classic monotonic concession protocol (MCP) [11, 5, 10]). In our solution the MCP protocol has been altered slightly so that the social welfare (instead of the product of the individual agents utilities) in the group of UAVs is optimized. We have extended this algorithm to support more than two UAVs when iteratively solving multiple collisions. The *iterative peer-to-peer collision avoidance* (IPPCA) algorithm [8] is used as a reference solution to which the *multi-party algorithm collision avoidance* (MPCA) is compared. The IPPCA algorithm is an extension of pair optimization for multiple collisions among several UAVs based on utilities provided by themselves. The basic version provides a solution for a pair of colliding airplanes, see Figure 1. The algorithm optimizes social welfare in that pair, thus the aircraft would like their flight plans to maximize the sum of their utilities, but still find collision-free paths.

The implementation and the experiments have been carried out within the framework of the Air-Traffic Control (ATC) system – a multi-agent system for UAV operation simulation [8]. Each UAV is controlled by an autonomous agent. Besides other functions the agents implement collision detection and collision avoidance support to the components responsible for the flight path planning and execution. The implemented *cooperative* collision detection algorithms are based on sharing the local future flight plans among local UAVs by means of *subscribe-advertise protocol*. The system is built on top of $\mathcal{A}$-**globe** – the JAVA multi-agent platform [7]. $\mathcal{A}$-**globe** provides precise flight modelling of a large number of autonomous aircraft, flight plan planning, but also different geographical models and a human-machine (HM) interface. trajectory including time information.

Although the presented mechanism has been applied to autonomous aerial vehicles, the presented concepts can be

0-7695-3027-3/07 $25.00 © 2007 IEEE
DOI 10.1109/IAT.2007.58

used also in other domains requiring autonomous collision avoidance, e.g. autonomous ground vehicles (UGV). The collision avoidance in the field of UAVs is more complicated due to the fact that aircraft mostly cannot stop and stay at the same place before collision is solved.

2 Multi-Party Collision Avoidance

The *multi-party collision avoidance* (MPCA) approach removes the iteration known from IPPCA algorithm during multi-collision situation. It introduces a multiparty coordinator who is responsible for state space expansion and searching for optimal solution of multi-collision. A *multi-party coordinator* is an agent whose role is to find a set of flight plans of UAVs in the multi-party group which will not be mutually colliding. The *multi-party group* is a set of UAVs whose flight plans in state space are colliding with another flight plan from the group. The coordinator holds information about the group, state space tree, chooses which airplane will be invited to the group, requests UAVs in the group for generating evasion of a collision or sends the information about found non-colliding flight plans. An *evasion* is a changed flight plan after application of avoiding maneuver to the place of detected flight plan collision. Note that MPCA algorithm is running while planes are flying – time for solving the collision is limited.

A coordinator agent is created by the master plane of the pair which detects a future collision in their current flight plans (cooperative detection is described in [8]). A master in the pair is determined according to an alphabetical order of UAV's IDs. The coordinator is started at the master UAV. When a coordinator is created it starts to search for a non-colliding set of flight plans. The collision which causes to create a new coordinator is then treated as a *initial collision*. The searching algorithm proceeds in three successive steps which are repeated until solution is found.

Step 1 – The coordinator asks 2 planes from the group for possible flight plan changes to avoid their collision. The selection process is described in the next section. Each of these planes individually generate a set of evasions and send them back to the coordinator. The coordinator doesn't receive entire flight plans, but works only with its limited future part and the plans are identified by unique IDs in the communication between coordinator and participating UAV. E.g. coordinator requests to generate evasions for a particular flight plan identified by an ID at a particular collision time – the first and the last point of collision [4]. Additionally the plane checks for each evasion whether it is colliding with any other plane in the flight space and adds a notification about it to the evasion.

Step 2 – When the coordinator receives evasions, it expands the state space by making their cartesian combination. The UAV which generates possible evasions also checks if they are colliding with other airplanes not included in the coordination group and adds this collision flag to the response. Then the coordinator decides whether it will invite a new UAV that is not in its group yet and is colliding with received evasions, to enter the group. In the current version the size of the coordination group is not restricted.

Step 3 – The coordinator searches through combinations of generated flight plans (evasions). If the coordinator decides that more evasions are needed, it continues with step 1. If it finds the final solution (a set of non-colliding flight plans), it sends a message to planes from the group with their new flight plan.

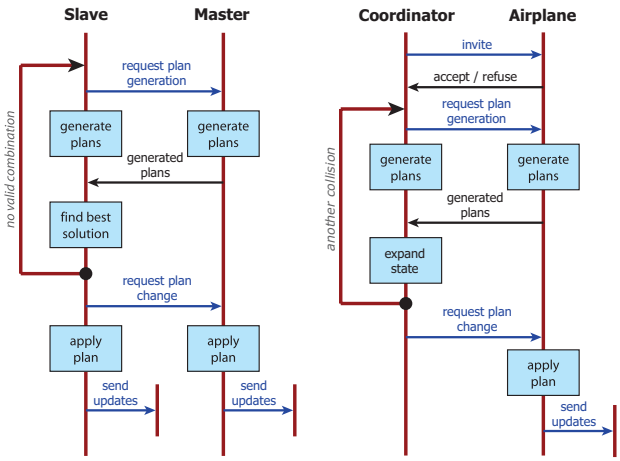

Figure 1. The negotiation during IPPCA (left) and MPCA (right).

The main part of a communication during a session between the coordinator and a UAV is shown in the Figure 1. As in IPPCA, the requests for plan generation are repeated according to the needs of the searching algorithm and state expansion. Additionally, a *failure message* can be sent in both directions. The failure signalizes that a session has to be terminated. It is used for example when the flight plan of the plane is changed for a reason unrelated to the group. A good example of a failure message usage is a situation when the multi-layer collision architecture [8] is switching between different collision solvers.

A coordinator assumes that the initial plans in the group are not changed during searching. If it does, the part of searching based on the old flight plan has to be omitted – by removing the UAV with changed flight plan from the group. This plane can be added to the group again if its new flight plan is colliding with planes from the group. This relates to the problem of concurrent existence of several coordination multi-party groups as described in the section 2.1.

The searching algorithm is based on A* algorithm [6] where the heuristic is zero – described below. The algo-

rithm expands the state with the best current utility value (implemented by priority queue) which corresponds to its function to maximize social welfare. MPCA algorithm is the same as IPPCA method for multi-party groups with two UAVs in the final state. The algorithm with zero heuristics finds the best possible solution, however the expanded state space grows exponentially with the number of collisions in a multiparty group. Therefore we have also implemented a second not admissible heuristic called "collision" which is used in the situation when the state expansion would exhaust computer resources.

2.1 Interaction of Multi-party groups

As described in section 2, a pair of colliding airplanes can create a coordinator. More pairs of planes can identify different collisions at the same time. Concurrent existence of more coordinators and multiparty-groups is then inevitable.

In our implementation we've allowed existence of several coordinators at the same time and if they share some of the UAV, the situation is handled by the used *collision solver manager* [8]. A plane can be contained in more multiparty groups, but it can be actively participating only in the group with the highest priority – mainly given by the time of group's initial collision (described above). The more imminent collision has higher priority. If the time of initial collision is the same, they are ordered according to the alphabetical order of the coordinator ID. All UAVs participating in the same multi-party groups have the same priority order of coordinators. This prevents the solving deadlocks – there will be ever at least one coordinator which will be treated as the most important and his requests will be served. A plane creates a coordinator iff it is not in a group which has a more important coordinator. This rule assures that there will be ever a coordinator for globally most important collision and prevents a redundant creation of coordinators.

When some coordinator successfully finishes, the planes in its group change their flight plans and send failures to other groups they participate in. This way the other coordinators know that they cannot use old flight plans of that planes in their state space – they remove part of their state space which was expanded with the use of the old non-existing flight plan and in all states flight plan is removed.

3 Experimental Evaluation

A set of tests were performed where several characteristic properties were collected and then used for comparing both collision avoidance methods. Experiments have been carried out within the limited square area of 31 x 31 units. The sequence of 900 runs (configuration with 5, 10, 15 ... 90 UAVs, each run **repetitively 50 times** to provide average result values) has been measured for both methods. The UAVs' start and destination way-points are randomly generated on the two opposite borders of a square area, thus each UAV needs to fly across the square. All way-points are generated at the same altitude during the whole experiment. Way-points of each new UAV are generated on adjacent borders of the square in clock-wise direction. Thus high number of collisions is guaranteed as a result of using this generation method. The radius of each UAV's safety zone is 0.25 units and its radar range radius is 10 units. The UAV's flight speed can vary between 0.075 and 0.125 units per second with acceleration and decelaration of 0.05 speed units per second. Totaly 85500 UAVs were simulated during more than 230 hours of the flight time. A collision-free solution was found in each run – all UAVs respect the safety area around all others during the simulation.

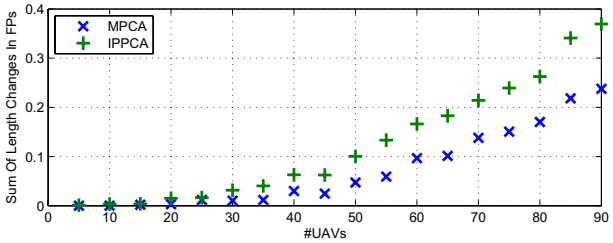

Figure 2. The sum of differences between final collision-free paths and shortest regardless collisions.

The Figure 2 presents the comparison of the average sum of all differences between the final collision-free flight plans and the shortest path from start to end way-points (euclidian distance) in given run. The results validate benefits of the MPCA algorithm to provide a more optimal solution – depending on the number of UAVs, the results are improved by 10 to 50 percentage compared to the IPPCA. The value varies due to the fact that the same number and types of collisions during each randomized experiment is not guaranteed.

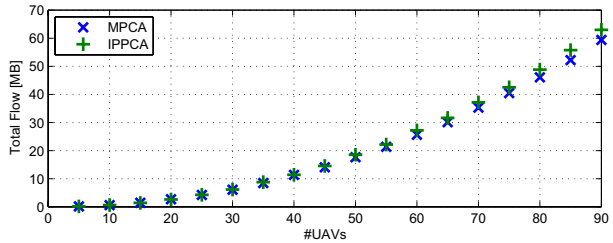

Figure 3. The communication flow analysis.

We've studied communication aspects of both algorithms, see Figure 3. Both algorithms have almost the same amount of transmitted bytes, but the difference is in the flow distribution during the experiment run. Again the chart presents the average value derived from **50 repeated** tests. We observed that the MPCA requires several times wider communication bandwidth than IPPCA, especially for the runs with more UAVs.The MPCA requires more communication during the state expansion phase within the coordination group but on the other hand it requires smaller number of coordination groups due to the fact that MPCA solves multi-collision in one change. This leads to almost the same sum of total bytes transmitted among all UAVs in the run.

4 Conclusion

In the paper we present a novel approach addressing the problem of distributed cooperative collision avoidance among agent-controlled autonomous unmanned aerial vehicles (UAVs). The *multi-party collision avoidance* (MPCA) algorithm is described and compared to the *iterative peer-to-peer* (IPPCA) approach. The algorithm utilizes the benefits of a centralized solution-searching and optimization but still decentralized to the air vehicles. Several UAVs which have mutual collision are grouped together and create a *coordinator agent* which is responsible for the state space generation and searching for the optimal solution. It can invite new UAV to the group if it is necessary or also negotiate with other existing group to merge them into one.

A set of experiments has been carried out and several algorithm aspects were studied. In these experiments we validate that the MPCA algorithm doesn't provide worse solution than IPPCA in any configuration and gives much better solution for more UAVs with several multi-collision situations among them. In our next work we would like to focus on the situations when there is only a limited communication range available to the UAVs. We are planning to implement a group size limit and define a membership function which will help to identify which UAV to replace by a new one that needs to become a part of the same group. Such restriction will still provide a solution close to optimum but requires less computation power than the current unlimited version.

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[1]. Also supported by Czech Ministry of Education grant 6840770038.

References

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

[2] 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.

[3] 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.

[4] M. Pěchouček, D. Šišlák, D. Pavlíček, and M. Uller. Autonomous agents for air-traffic deconfliction. In P. Stone and G. Weiss, editors, *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1498–1505. ACM, 2006.

[5] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter*. The MIT Press, Cambridge, Massachusetts, 1994.

[6] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artificial Intelligence, Englewood Cliffs, New Jersey, 1995.

[7] 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.

[8] 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.

[9] J. P. Wangermann and R. F. Stengel. Optimization and coordination of multiagent systems using principled negotiation. *Journal of Guidance, Control, and Dynamics*, 22(1):43–50, 1999.

[10] S. Wollkind, J. Valasek, and T. R. Ioerger. Automated conflict resolution for air traffic management using cooperative multiagent negotiation. In *Proc. of the American Inst. of Aeronautics and Astronautics Conference on Guidance, Navigation, and Control*, Providence, RI, 2004.

[11] 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.

[1]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.