43rd IEEE Conference on Decision and Control
December 14-17, 2004
Atlantis, Paradise Island, Bahamas

Automated Multiple UAV Flight – the Stanford DragonFly UAV Program

Rodney Teo, Jung Soon Jang, Claire J. Tomlin
Dept. of Aeronautics and Astronautics, Stanford University, CA 94305
trodney,jsjang,tomlin@stanford.edu

***Abstract*— The Stanford DragonFly Program (http://airtraffic1.stanford.edu/∼uav/) consists of two fixed wing (10-foot wingspan) autonomous aircraft, whose onboard architecture, avionics, control, and integration with wireless communications have been developed entirely at Stanford. The DragonFly platform is used as a technology testbed for research in the design of a decision-theoretic real-time operating system, and to experimentally validate hybrid control of single vehicles, and real time danger zone computation and avoidance for two automated vehicles flying in parallel. Since June 2003, we have been successfully flying the two automated DragonFly aircraft simultaneously, in dual vehicle approach patterns, and have demonstrated our automatic collision avoidance system on this platform (one aircraft 'blunders' into the path of the second, the other invokes an 'emergency escape maneuver' based on the relative configuration of the two aircraft). In June 2004, based on the success of our DragonFly tests, we flew our software on a Boeing test platform of two aircraft: an F-15, and a T-33 test aircraft.**

I. Introduction

Coordination and control of unmanned aerial vehicles (UAVs) [1], [12], [14], [2], [11] has been a growing research area with strong support from both civilian and military applications, yet we are just beginning to understand the potential impact of UAV operations on 21st century aviation. At Stanford, we designed and built the DragonFly UAVs, as shown in Figure 1. They are fixed wing, 10 foot wingspan, fully automated aircraft. This platform has been a wonderful testbed to support and validate our research on applied advanced control: these two vehicles are providing the experimental testbed for single vehicle control and navigation [6], [7], and dual vehicle coordinated approach control [13], and will be used as part of a larger testbed for our research on decentralized optimization schemes for formation flying [3].

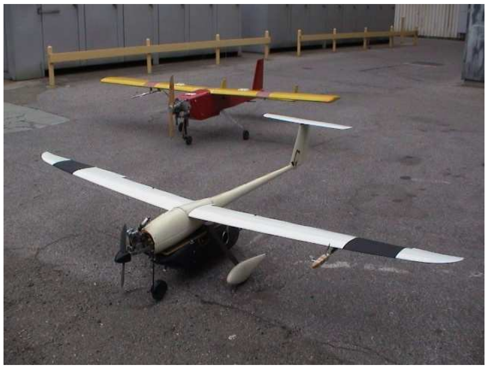

Fig. 1. A photograph of the Stanford DragonFly UAVs (DF2 and DF3). The aircraft are approximately 10 foot in wingspan, with onboard navigation, communication, and control algorithms, all designed by us.

In this paper, we discuss the design and implementation, on the DragonFly UAVs, of one of our multiple vehicle control algorithms. Specifically, we present an algorithm for collision avoidance, given that one vehicle, the *blunderer*, may blunder into the path of the other, the *evader*. The algorithm is based on differential game theory, which is used to guarantee that threshold separation distances can be maintained regardless of the blunderer's and evader's moves. We consider a two player pursuit-evasion game, in which the blunderer is modelled as the pursuer, and the evader is modelled to perform a nominal emergency escape maneuver (EEM), with allowed bounded perturbations from the nominal. Contrary to a pursuit-evasion game, however, the evader is assumed to "play" the perturbations in order to assist the pursuer in realizing its objective of minimizing the distance between the two aircraft. Solving this problem will give the conditions under which the evader is able to successfully conduct the EEM to maintain separation from the blunderer, regardless of the blunderer's actions and regardless of the evader's perturbation about its nominal EEM. This algorithm is particularly relevant for safety critical control of two aircraft, when the goal is to find a safe, conservative solution to uncertain actions of the two aircraft. We have applied this algorithm to Closely Spaced Parallel Approaches (CSPA) of aircraft into airports. The algorithm and its application to CSPA, is presented in detail in [10], [13].

The implementation of this algorithm on the Stanford DragonFly UAVs requires full automation and trajectory tracking (under uncertainty) capabilities for these UAVs. Thus, we also briefly outline in this paper the underlying technology which we designed to provide a robust, safe, and high performance UAV control system. The algorithm combines nonlinear tracking with a control input limiting algorithm (CILA) to prevent actuator saturation. These algorithms are presented in detail in [5]. We conclude with

This research has been supported by DARPA's Software Enabled Control Program, and by ONR through the Computational Methods for Collaborative Motion MURI.

0-7803-8682-5/04/$20.00 ©2004 IEEE

a discussion of our flight test results.

II. Dual Vehicle Collision Avoidance Using Game Theory

An Emergency Evasive Maneuver (EEM) is a prescribed maneuver for evading the blundering aircraft to avoid a collision. With pursuit-evasion game theory prescribing the best maneuvers to escape from the pursuer, it turns out that the best maneuver is different for different encounters with the pursuer. To make the EEM close to optimal, and yet to retain the simplicity of having a small number of EEMs to fly, or to program into the autopilot, we choose two EEMs: if the evader leads the pursuer, then the evader should accelerate, climb, and turn to 45 deg; if the evader trails the pursuer, then the evader should maintain speed, climb, and turn to 60 deg. The climb maneuver is included to gain altitude (for CSPA, this is necessary in order to get to the missed approach altitude for a "Go Around" maneuver), it is not needed to avoid the blundering aircraft. The aircraft are assumed to be subject to translating sideways in addition to forward, taking into account the effects of cross winds and gusts. It is also assumed that the evader is unable to fly the prescribed EEM precisely, and that it will fly some trajectory which is within certain bounds of the prescribed EEM.

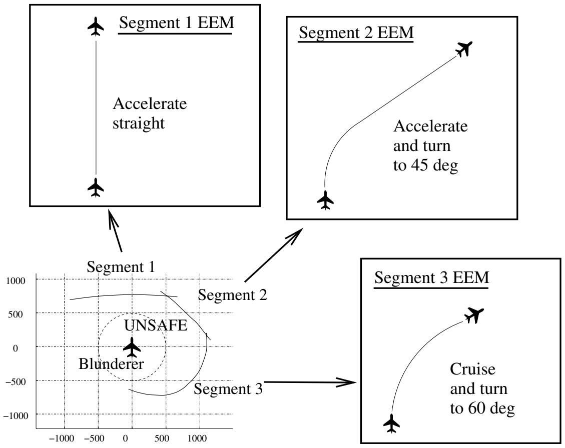

Fig. 2. A sample danger zone result generated using 3 different EEMs.

Before presenting the algorithm, a useful way to see the value of this algorithm is in terms of the resulting danger zones it forecasts and avoids. An example of a danger zone is shown as the "UNSAFE" region in Figure 2. Segments 1 to 3 are obtained using the EEMs labeled 1 to 3 respectively, in Figure 2. These EEMs are also shown in the figure. The danger zone is a region in which an EEM that is initiated inside it is not guaranteed to be safe. An EEM initiated outside or on the boundary of the danger zone, on the other hand, is guaranteed to be safe provided that the initiated EEM is the appropriate one. The appropriate EEM is the one that is used to generate the boundary that is being encroached. For example, if the evader is encroached by Segment 2 of the boundary, then the appropriate EEM to conduct is the Segment 2 EEM. In fact, so long as the Segment 2 EEM is conducted outside Segment 2 of the boundary, that EEM is guaranteed to be safe. The danger zone is not static. It is computed in real-time and its size and shape is a function of the aircraft speeds, relative heading, and worst-case bounds on the variation of their speeds and turn rates from the nominal. In this paper, the results presented are for an evader that is performing the approach on the right of the blunderer. Thus, the turns in the EEMs above are turns to the right (away from the blunderer). The results for an evader performing the approach on the left is simply a mirror image of the results shown here.

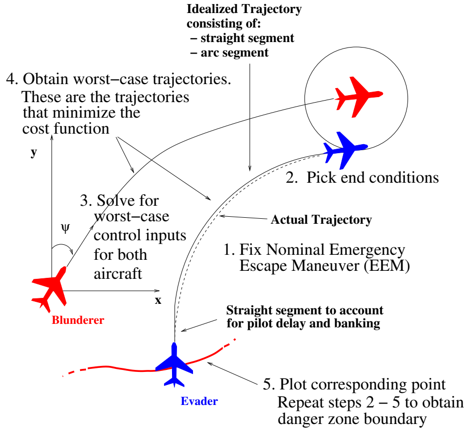

Fig. 3. Overview of the Approach.

The problem is to compute the boundary of the danger zone that guarantees safety when the appropriate EEM is conducted. The danger zone boundary consists of two segments (Segments 2 and 3) as discussed in the previous section. (A third segment, Segment 1, is included only for illustration purposes.) The computation of each segment is performed separately and then the resulting three segments are combined. The approach for computing the boundaries of each of the three segments is the same. A solution overview for one segment is as follows (Figure 3):

1) Consider a nominal EEM (for the segment that is to be computed) as an idealized trajectory consisting of a straight segment (to account for the pilot response delay, and for the time lag to achieve the required bank angle and acceleration) and an arc segment generated by the constant nominal turn rate of the EEM. The nominal EEM is thus a conservative approximation to the actual trajectory of the evader: The uncertainty bounds in forward and lateral speed, and the turn rate of the evader, are chosen so that the actual trajectory will be contained within this representation.
2) For this EEM, determine the end conditions of the computation, which are the relative position and orientation between blunderer and evader (terminal states) when a collision has just occurred. Recall that a collision is defined to occur just when the separation distance equals the threshold of 500 ft.

3) Determine using optimal control techniques the worst-case control inputs required of both the blunderer and evader to drive the relative position and orientation between blunderer and evader to these end conditions, and minimize one of the following cost functions depending on the solution sought:
   a) the time to collision (game of degree) or,
   b) the separation distance at termination (game of kind).
4) Compute (analytically, using these worst-case control inputs) the corresponding blunderer and evader state trajectory.
5) Obtain the danger zone boundary from the analytical solutions of the state propagation by repeating Steps 2 to 5 over the space of permissible end conditions.

In Step 3, we refer to the worst case control inputs, so named because any other choice of control inputs within the allowed bounds can only improve the outcome of the game from the evader's perspective. As opposed to the min-max solution in a differential game problem [4], [9], a "min-min" problem is solved here, where the evader too tries to "assist" the blunderer in minimizing the cost function. This is used to guarantee safety even in the face of uncertainty in the model and maneuver, with respect to the evader.

The details of the algorithm, and all of the calculations, are presented in [10], [13]. We have computed the danger zones for three different cases; in this paper we present the results for the Baseline case, which is shown in Figure 4. In the Baseline case, the speeds and turn rates of the blunderer and evader are similar. The parameters of the Baseline case are shown in Table I. Speed and acceleration are in knots and knots/sec respectively. Turn rate is in deg/sec and the "delay" is the time in seconds between the start of the blunder and the start of the evader's acceleration and turn.

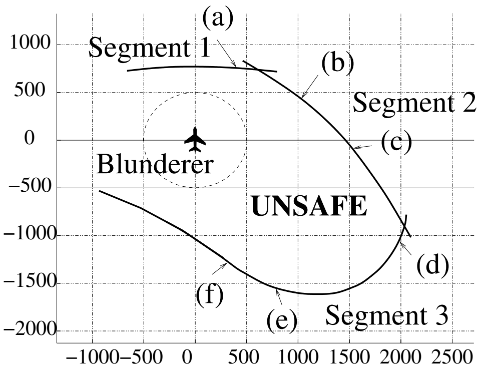

Fig. 4. Danger Zone boundaries: Baseline case (ft).

For the Baseline case, the corresponding worst-case trajectories that generated the points (a) through (f) on the boundary in Figure 4 are shown in Figures 5 to 10. These are the result of the kinematic representation of aircraft motion, which conservatively approximates the motion of real aircraft. A switch in v will be termed a heading switch. We can now attempt to understand intuitively these computed solutions.

TABLE I

BASELINE CASE PARAMETERS (SPEEDS ARE GIVEN IN KNOTS, TURN RATES IN DEG/SEC, ACCELERATION IN KNOTS/SEC AND DELAY IN SEC).

u_b	155 - 165	u_e	150 - 160	a_e	1
v_b	+/-8	v_e	+/-8	**delay**	3
ω_b	+/-4.5	ω_e	4.5+/-0.5		

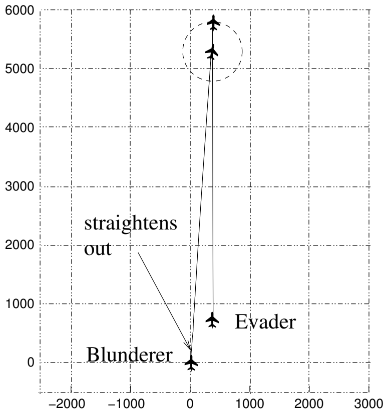

Fig. 5. Point (a) baseline case: Evader accelerates straight ahead (ft).

Figure 5. In this case (point (a) in Figure 4), the worst-case trajectory for the blunderer is to turn to the right slightly and fly straight as fast as it can. It is not obvious that it is better for the evader to accelerate straight ahead, rather than to accelerate and turn to the right by 45 deg. The overall analysis confirms the former to be so, as we can see that point (a) in Figure 4 is on the unsafe side of Segment 2 of the boundary. Recall that Segment 2 assumes that the evader accelerates and turns right by 45 deg.

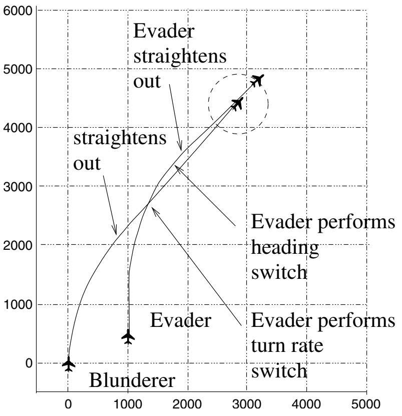

Fig. 6. Point (b) baseline case: Evader accelerates and turns 45 deg (ft).

Figure 6. This case (point (b) in Figure 4) is interesting because the trajectory of the evader involves control input switches. Upon crossing the blunderer's "future" worst-case

path, the evader switches from turning as slowly as it can to turning as fast as it can. This is logical as when the evader is to the right of the blunderer, by turning as slowly as it can, it gets itself nearer to the blunderer's path. When the evader is to the left of the blunderer, by turning as fast as it can, it gets itself back towards the blunderer's path. In addition to the turn-rate switch, there is a heading switch when the heading of the evader is parallel to the straight part of the blunderer's trajectory.

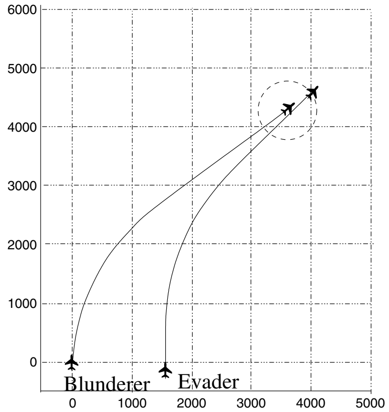

Fig. 7. Point (c) baseline case: Evader accelerates and turns 45 deg (ft).

Figure 7. In this case (point (c) in Figure 4), the evader again accelerates whilst turning right up to 45 deg. However, the evader remains on the right side of the blunderer. Because of this, the worst-case evader trajectory does not involve employing the turn rate and heading switches.

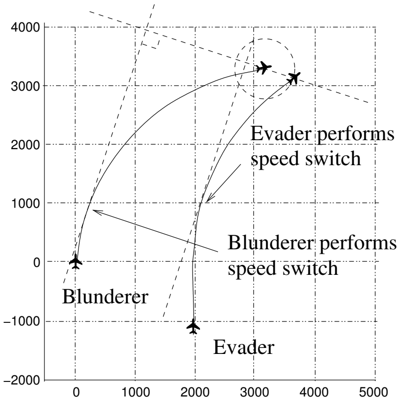

Fig. 8. Point (d): Evader turns 60 deg without acceleration (ft).

Figure 8. Note that this case (point (d) in Figure 4) is on Segment 3. It involves forward speed switches for both aircraft. The blunderer initially flies as slowly as it can and then as fast as it can when its heading is perpendicular to the line joining the two aircraft at game termination. This seems logical because it is choosing its speed depending on whether it is pointing in the same general direction as the line from the blunderer's terminal state to the evader's terminal state. The speed switch of the evader can be similarly explained.

Figure 9. In this case (point (e) in Figure 4), the blunderer flies as slowly as it can throughout. It switches its heading from left to right at the point where its heading is parallel to the line joining the two aircraft at termination. The evader flies as fast as it can throughout and switches its heading from right to left at the point where its heading is parallel to that same line.

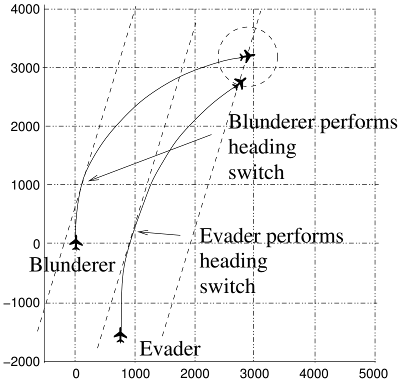

Fig. 9. Point (e): Evader turns 60 deg without acceleration (ft).

Figure 10. In this case (point (f) in Figure 4), there are turn switches in addition to the heading switches. For the blunderer, it turns at its maximum turn rate to the left initially and then switches to its maximum turn rate to the right. This has the effect of slowing down its motion in the direction of the line joining the two aircraft at termination. The evader, on the other hand, turns as quickly as it can, and then switches to turning as slowly as it can. Because the evader ends somewhat behind the blunderer, these behaviors tend to minimize the distance between the two aircraft at termination.

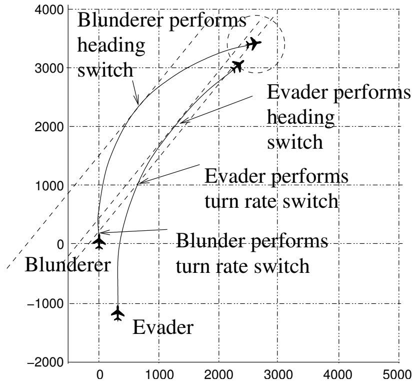

Fig. 10. Point (f): Evader turns 60 deg without acceleration (ft).

In our implementation, the algorithm runs on one aircraft. It takes speed and heading information from both aircraft and computes the danger zone polygon, which is a guaranteed overapproximation of the zone computed above, and easier and faster to implement. With the position information from both aircraft, it then performs the check of whether the boundary of the danger zone encroaches the position of the own aircraft. If so, it activates the necessary EEM. During the EEM, speed and heading information of the blundering aircraft will not be required unless they change beyond the limits expected. Full in-flight experimental validation will be shown in Section 4.

III. Automatic control of the DragonFly UAVs

We have developed both high fidelity nonlinear and linear models of the DragonFly 2 (DF2) aircraft; and have designed control algorithms for both models. Here, we describe some of the linear design that was used in the flight tests of the next section. Both schemes are presented in detail in [5].

Perturbed linear models for both longitudinal and lateral dynamics were obtained from a system identification technique using the data from flight tests. For flight tests, maneuvers were designed to excite natural modes of the aircraft (although the design relies on *a priori* knowledge of the aircraft dynamics). A ground pilot performed these maneuvers through an R/F link, and the resulting flight data were collected by the avionics. System identification was then carried out using a least-squares method in the frequency domain [8], [5]. The identification cycle (from maneuver design to analysis of flight data) was repeated until acceptable results were obtained.

We used standard time-invariant longitudinal and lateral aircraft dynamics, and after system identification, a parameter set was determined which resulted in close correlation between simulated and actual data [5]. Here, we show in Figure 11 input and output time histories measured in flight for the 3-2-1-1 elevator and pulse throttle and 2-1-1 rudder and aileron, at the flight condition, V_T=30.14 m/s and α_e = -1.95 degrees. The designed input form was distorted somewhat by the pilot and the servo controller, as can be seen in the figures.

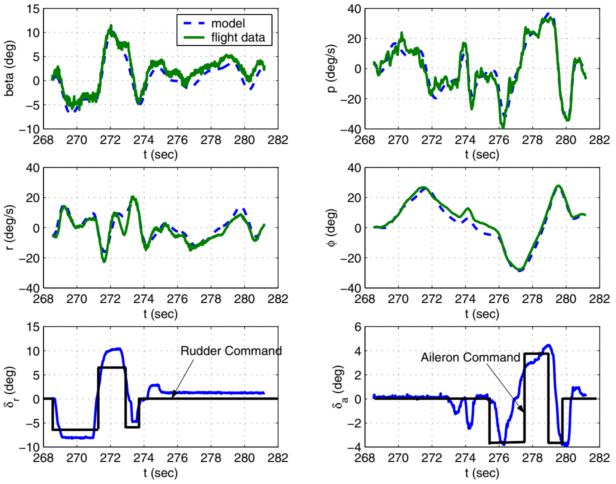

Fig. 11. Lateral dynamics identification result for 2-1-1 input.

We have designed linear control laws based on these dynamics, for several flight modes of interest, including those which implement the EEMs of the previous section. In addition, we have developed and implemented a real time algorithm to avoid actuator saturation [5]. In our scheme, we modify the reference dynamics by effectively decreasing its bandwidth, whenever its magnitude lies outside of the boundary of an maximum allowable computed envelope. This envelope is a mapping of the maximum input deflection boundary imposed (by an actuator or a designer), through the dynamics of the system. For a given value of the state, the mapping is always linear if the plant dynamics is bilinear (linear to control inputs). Therefore, as long as the reference dynamics lies inside of the maximum allowable computed envelope, the control doesn't saturate. Intuitively, this can be explained as follows: actuator saturations tend to imply that gain/bandwidth of the resulting closed-loop system is too high; conversely, if it is known that actuators are going to saturate, then the bandwidth should be decreased. The proposed scheme is simpler in structure and can be computed quickly in a control loop so that it is applicable in real-time: it does not require the redesign of a feedback controller and/or the augmentation of additional control loops to satisfy given actuator constraints. Also, such a formulation allows the expression of a sufficient condition of Lyapunov stability of the closed-loop system under saturation.

IV. Flight test results

With both aircraft automated to fly in heading and altitude hold modes, blunder and EEM, the test bed was ready for the full in-flight experimental demonstration of the algorithm. Figure 12 shows the experiment being conducted over Moffett Federal Airfield. Over thirty runs were made with DF3 blundering and DF2 performing the EEM as triggered by the danger zone computation algorithm. All runs were completely successful. Sample results are shown in Figures 13 and 14. The separation distances are greater than the threshold which is scaled down to 30m for the experiment.

Fig. 12. Preparation for In-Flight Experimental Demonstration.

Note that the separation threshold is never violated. The flight demonstration shows the implementation of the

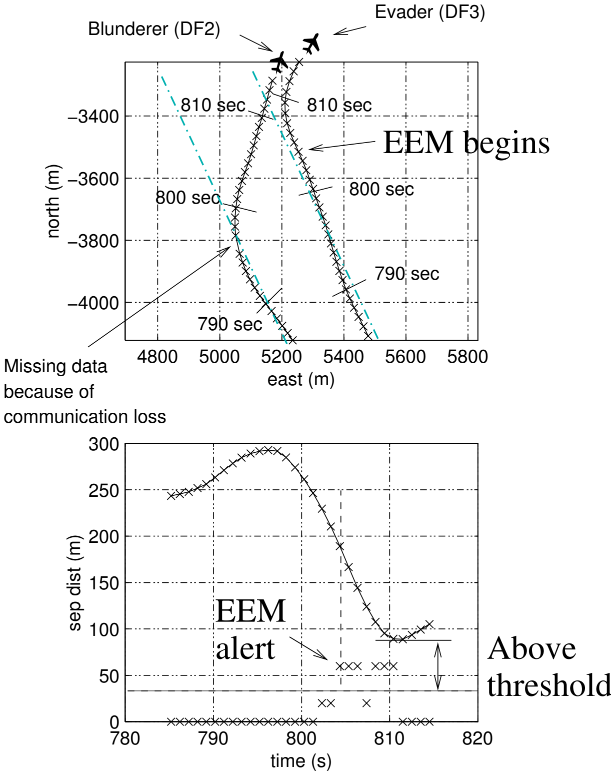

Fig. 13. Experimental Flight Demonstration: Independent-Approach Scenario. Accelerate and Turn to 45 deg EEM alert given. (The 'x' plot indicates the output of our collision avoidance algorithm.) The threshold of 30 m is the scaled-down equivalent of 500 ft to go with 10 foot-wing-span aircraft used in the flight experiments.

algorithm and in particular, demonstrates that it can run in real-time.

V. Conclusions

A game theoretic approach for computing, in real time, danger zones in two-aircraft collision avoidance problems has been presented in this paper. Compared to other algorithms, this approach enables a more extensive range of blunder trajectories and variations in the evader emergency evasive maneuver to be considered and to be evaluated definitively. Implementation on the Stanford DragonFlies is discussed, and flight test results shown. This algorithm has applications to both UAV guidance, as well as to air traffic problems.

References

[1] Chandler, P.R. and Pachter, M. Research Issues in Autonomous Control of Tactical UAVs. *Proceedings of the American Control Conference*, June 1998.

[2] Frazzoli, E., Dahleh, M.A., and Feron, E. A Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopters. In *IEEE Conference on Decision and Control*, Phoenix, AZ, September 1999.

[3] Inalhan, G., Stipanovic, D., and Tomlin, C.J. Decentralized Optimization, with Application to Multiple Aircraft Coordination. In *Proceedings of the 41st IEEE Conference on Decison and Control*, Las Vegas, NV, March 2002.

[4] R. Isaacs. *Differential Games*. Robert E. Krieger Publishing Company, Huntington NY, 1975.

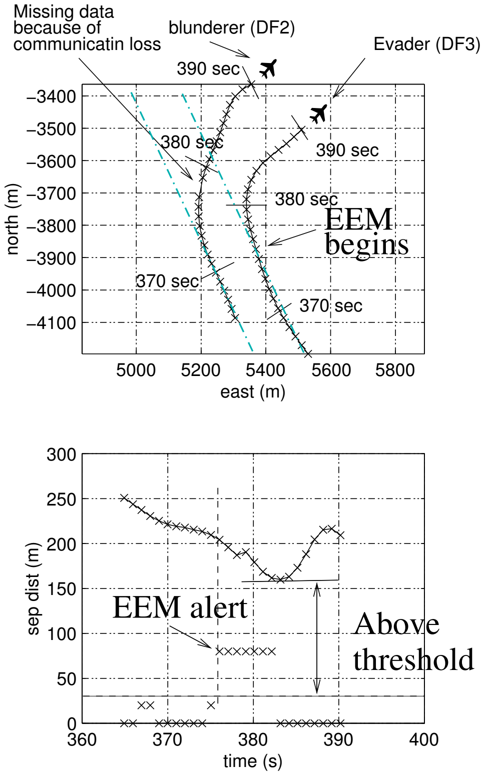

Fig. 14. Experimental Flight Demonstration: Dependent-Approach Scenario. Cruise and Turn to 60 deg EEM alert given.

[5] J. S. Jang. *Nonlinear control using discrete-time dynamic inversion under input saturation: Theory and Experiment on the Stanford DragonFly UAVs*. PhD thesis, Stanford University, 2003.

[6] Jang, J.S. and Tomlin, C.J. Autopilot Design for the Stanford DragonFly UAV: Validation through Hardware-in-the-Loop simulation. In *Proceedings of the AIAA GNC Conference*, Montreal, Canada, August 2001.

[7] Jang, J.S. and Tomlin, C.J. Longitudinal Stability Augmentation System Design for the DragonFly UAV Using a Single GPS Receiver. In *Proceedings of the AIAA GNC Conference*, Austin, TX, August 2003.

[8] Morelli, E.A. Real-time Parameter Estimation in the Frequency Domain. In *Proceedings of the AIAA GNC Conference*, Portland, OR, 1999.

[9] U. R. Prasad, N. Rajan, and N. J. Rao. Planar pursuit-evasion with variable speeds, part 1, extremal trajectory maps. *Journal of Optimization Theory and Applications*, 33(3):401–418, 1981.

[10] R. Teo. *Computing Danger Zones for Provably Safe Closely Spaced Parallel Approaches: Theory and Experiment*. PhD thesis, Stanford University, 2003.

[11] Shim, D.H., Kim, H.J., and Sastry, S. Hierarchical Control System Synthesis for Rotorcraft-based Unmanned Aerial Vehicles. In *AIAA Guidance, Navigation, and Control Conference*, Denver, CO, August 2000.

[12] Singh, S.N. Adaptive Feedback Linearizing Nonlinear Close Formation Control of UAVs. *Proceedings of the American Control Conference*, June 2000.

[13] Teo, R. and Tomlin, C.J. Computing Danger Zones for Provably Safe Closely Spaced Parallel Approaches. *Journal of Guidance, Control and Dynamics*, 26(3):434–442, May-June 2003.

[14] Verma, A., and Subbarao, K., and Junkins, J. A Novel Trajectory Tracking Methodology using Structured Adaptive Model Inversion for Uninhabited Aerial Vehicles. In *Proceedings of the American Control Conference*, Chicago, June 2000.