2007 Information, Decision and Control

Effect of Uncertainties on UCAV Trajectory Optimisation Using Evolutionary Programming

Istas F. Nusyirwan [1], Cees Bil [2]
The Sir Lawrence Wackett Centre for Aerospace Design Technology, RMIT University
GPO Box 2476V, Melbourne VIC 3001, Australia,
[1] *s3093201@student.rmit.edu.au* [2] *cees.bil@rmit.edu.au*

Abstract

There is a recognised need for an automated trajectory planning to guide manned or unmanned aircraft against an agile adversary such as a missile. Evolutionary programming approaches provide an alternative to classical functional optimisation methods with the capability of incorporating multiple optimisation goals and in the same time tolerating aircraft constraints. In this study, an evolutionary flight path planning algorithm capable of mapping aircraft trajectories in three dimensions under several aerodynamics constraints is developed. The task of the trajectory was to guide the aircraft away from interception. The calculations assumes that the aircraft states are accurate. Good trajectories were found under this assumption. But in reality, states are measured in an environment that has uncertainties, such as instrument error, atmospheric disturbances, etc. This paper studies the effect of the presence of errors to the accuracy of the algorithm. Two state variables were studied, i.e. altitude and velocity for both players. From the simulation, the effect of noises and interception radius can influence the sensitivity of the optimiser.

1. Introduction

A two player pursuit-evasion game between is a problem in which they strive for a common performance index. One player (the pursuer) wants to maximise it and the other player (the evader) wants to minimise it [7]. It is called differential games when the games are expressed in ordinary differential equations. The study of pursuit-evasion differential games have been pursued by many such as [1],[2] and [4]. The results from this approach, although has attracted a considerable interest, seem to be difficult for actual application [7].

[7] has suggested several ways to overcome this problem. The basic idea is to give the players *a prior* optimal or sub optimal feedback strategies. These strategies are evaluated by conducting massive simulations in the parameter space of initial geometries and guidance law parameters, and analysing the results. Good solutions were found although, statistically the probability is very small.

[8] analysed differential game problems consisting of two aircraft with variable speeds in coplanar motion (horizontal plane), i.e. 2-dimension. His objective is to solve realistic aerial combat problems. There parameters were important for longitudinal acceleration, i.e. speed, turn rate and throttle setting. The pursuer uses throttle setting and turn rate as the control input, whereas the evader uses only turn rate as the control input. A modified differential dynamic programming method is used as the optimisation algorithm for solving optimal open-loop differential games.

Another approach of solving pursuit-evasion games is by the descretisation of optimal control. Two methods were proposed by Tuomas [6]. One is the solution of the necessary conditions of the continuous-time game is broken down into ordinary optimal control problems. These control problems can be solved using discretisation and nonlinear programming techniques. The second method is to discretised the game and transformed into a bilevel programming problem and solved using a first order feasible direction method. He demonstrated the solution using these methods between a realistically modeled aircraft and a missile at the end game using the terminal time as the payoff.

In this paper, a technique using evolutionary algorithm to search for optimal control for an evader against a much more agile pursuer is proposed. In this technique, an initial population of strategies for the evader were created in random. Using evolutionary programming, good strategies were found. However, in reality, the states of other aircraft has to be estimated due to sensor noise. The difference between the actual states and the measured states is called error. The idea is to introduce ”errors” in the optimisation algorithm and to see if these errors could influence the accuracy of the algorithm. In this study, we would like to see the sensitivity of the optimisation to altitude error and interception radius.

The paper is divided into 5 main sections. Section 2 discusses the methodology used in the algorithm. In this section, we discussed about the equation of motion used, the development of the evolutionary programming and the intelligence of the pursuer. Section 3 discusses several scenarios for the purpose of analysis and comparison. Section 4 discusses the results from each scenario. And finally, the conclusion is at section 5.

2. Methodology

A. Evolutionary Programming

Evolutionary programming is selected because it opens up the possibility to search for optimal solution with the presence of nonlinearity, parameter discontinuity and discrete input.

1-4244-0902-0/07/$20.00 © 2007 IEEE

Although the search is stochastic but still with the presence of powerful computer power, good solutions can be found in a relatively short time, i.e. around 10-15 seconds.

This algorithm gives the optimal path in three dimensions just as like the actual avoidance manoeuvre executed by an aircraft.

In this method, the search for optimal path begins by initially randomly generating a population of possible paths [3]. A path consists the information of heading angle change, flight path angle change and thrust setting for every time step.

Each member of the population (a solution) is evaluated and given a fitness value. The fitness value tells how good the solution is.

The evaluation of the solution is done by running a simulation for a period of time. In this case for 100 seconds. The simulation starts at a known initial states of both evader and pursuer at time, $t = 0$. The evader uses the path given by the tested solution and the pursuer uses its own guidance system to guide itself toward the evader. A solution is considered good if within the 100 seconds of the simulation, the evader manages to evade interception and, at the same time, does not go over the aircraft's aerodynamic and performance constraints. If the solution exceeds the aircraft's constraints, although successfully avoid interception, the solution is considered "not good" but still can be used to produce the next offspring. The reason is to avoid from being locked in the local optimal region during the optimisation cycle.

Recombination through mutation is perform after evaluation. Good solutions are retained and mutated in hoping it produces a much better solution. The mutated solutions are called the offspring. The offspring and the parent are combined together to be the next generation of the population. The population has to go through the evaluation cycle again.

The cycles are repeated until the number of maximum population has been reached. The best solution is the solution that has the highest fitness value.

B. Path Representation

In each population, there are 100 strategies or solutions. A strategy is actually an instruction for the aircraft to change its heading, flight path angle and throttle setting at every second. For example, at $t = 0$ s, the aircraft changes its heading angle by 20 degrees to the left, climb up by 15 degrees and set the throttle setting to 0.7, and at $t = 1$ s, again the aircraft has to change its heading angle, flight path angle to a new direction and the throttle setting to a new setting. The duration of the process is 100 seconds. The 100 seconds duration is chosen .

To represent a strategy in a computer program, the change of heading (ψ), flight path (γ) angles and the throttle setting, 1, has to be coded. This is made possible by determining the maximum permissible range for the heading, flight path angles and the throttle setting. In this research, the range of the angles is restricted between -15^0 to 15^0 for both heading and flight path angles, and the range of the throttle setting is between 0.2 and 1.0.

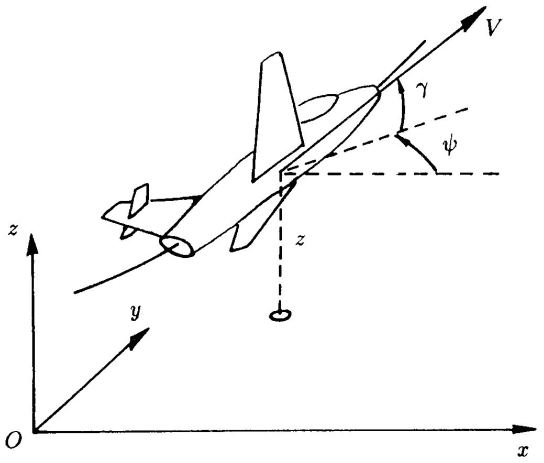

Fig. 1: Definition of heading angle (ψ) and flight path angle (γ)

TABLE 1: ENCODING THE HEADING ANGLE, FLIGHT PATH ANGLE AND THROTTLE SETTING

ID	Heading Angle Change (ψ), deg	Flight Path Angle Change, (γ), deg	Throttle Setting
1	-15	-15	0.2
2	-15	-15	0.3
3	-15	-15	0.4
⋮	⋮	⋮	⋮
5625	15	15	1

Discrete angle interval of 2.5^0 was used for heading and flight path angles, and for throttle setting the interval is 0.1. With this respect, we can now generate (24+1)(24+1)(9) = 5625 possible combinations of heading angle change, flight path angle change and throttle setting. Table 1 shows the coding of heading angle change, flight path angle change and throttle setting.

Instead of directly using the angles, the strategy uses the values of IDs as shown in Table 1. A series of numbers valued between 0001 and 5625 are randomly constructed such as shown in Figure 2 with 100 four-digits integer were

```
5314008457254247560509356125393127
1955963774513905975744073247895942
1638416360681628621833120095384540
4436240427587622650915052809305964
5611557127675775585105922016622128
0757734552447857411475502252475083
1043494624560883242707490555576948
1639956157217834864763179946572611
2331026042134938463258384168204356
1156784087365956750847441413904540
0877548131813438618358156040448261
2144056023599530052034156 8
```

Fig. 2: An example of coded path

ordered in series. The first value is 5314 means turn 13.75^0 to the left, cimb 3.75^0 and set the throttle to 0.5%. Next manoeuvre is 0084 which means 'and then turn 15^0 to the right, dive 3.75 and set the throttle to 0.4'. This is repeated for the next sequence up to the last sequence, i.e. 1568. The whole process is called the trajectory/path of the aircraft or a strategy. A population consists of 100 strategies. The number of population used in this study is set to 30.

C. Equation of Motion

In order to make the simulation as realistic as possible, aircraft equation of motion is applied. Three degree-of-freedom or point-mass aircraft model is used in this simulation. In this study, the trajectory of the vehicle at its center of mass (c.m.) is of greater interest than its attitude motions.

Newton's second law, aerodynamic and performance data are used for the simulation. We used the so-called Cartesian approach to simulation the aircraft's states [5]. The state variables are the vehicle's inertial velocity and inertial positions.

The derivation of the Cartesian approach is relatively straight forward. The inertial position and velocity coordinates are integrated from the Newton's second law. All the aerodynamic and the propulsion forces are fed into the Newton's second law equation given by Eq. 1.

$$m\mathbf{D}^I\mathbf{v}_B^I = \mathbf{f}_{a,p} + m\mathbf{g} \tag{1}$$

On the left side of Eq. 1 is the rotational derivative to the inertial frame I, with the body to inertial velocity V_B^I. The position of the aircraft in the inertial coordinate is found by integrating the velocity vector with respect to time.

The basic aerodynamic forces such as lift and drag are calculated based from the aircraft actual aerodynamic coefficient, C_L and C_D which are the function of altitude and Mach Number.

The propulsion force or thrust is a function of Mach Number, altitude and throttle setting. The thrust is modeled to be constant with airspeed and proportional to the air density as given in Eq. 2.

$$T_A = \tau\frac{\rho}{\rho_0}T_{A0} \tag{2}$$

where τ is a throttle setting [0-1], ρ is the air density at altitude, ρ_0 is the air density at standard sea level and T_{A0} is the full-throttle thrust developed in standard sea level.

The maximum turning rate is calculated by Eq. 3.

$$\dot{\psi}_{max} = \frac{g\sqrt{n_{max}^2 - 1}}{V} \tag{3}$$

D. Pursuer's Control and Guidance

Proportional Navigation Guidance system is employed by the pursuer. The navigation constant for in this study is set to be 4. The pursuer will use this guidance law throughout the game.

The pursuer's speed is governed through Newton's Second law. Thus the throttle setting, altitude, bank angle and flight path angle determine the speed.

TABLE 2: AIRCRAFT CONFIGURATION

Parameter	Role	
	Pursuer	Evader
Mass (kg)	6875	8500
Wing Area (m^2)	27.9	38
Wing Span (m)	9.1	11.4
$X_{initial}$ (m)	0.0	6000.0
$Y_{initial}$ (m)	0.0	0.0
$Z_{initial}$ (m)	5000	5000
$C_{L\alpha}$	1.1	1.3
C_{D0}	0.0412	0.0452
k	0.9	0.9
Thrust at Standard Sea Level (N)	160,000	180,000
Maximum N	9	9
Minimum N	-4	-4
Max Fuel Weight (kg)	3100	4000
TFSC (N/s/N	5.8×10^{-5}	6.0×10^{-5}
PNG Ratio	4	n/a

We assume that the pursuer always flies close to maximum speed, thus the turning is always governed by the aircraft structural limit.

The pursuer's initial states read by the evader are corrupted to a certain degree to simulate the error. The preceding pursuer's states are calculated and predicted on board the evader's computer.

The evader is assumed to know the pursuer's navigation guidance system and is using it in finding optimal trajectory to evade interception by the pursuer.

3. CASE STUDIES

We consider the problem by varying the error level and the interception radius. The error level selected is from 5% up to 25% from the actual states. At each error level, we would vary the interception radius. For each analysis, the number of trajectories without and with interception are observed and plotted. Table 2 shows the aircraft parameter studied in this paper. The maximum duration of the game is set to only 200 seconds.

The evader reads the pursuer's initial states and based from that, the evader optimises its trajectory. Good trajectories are found from the optimisation algorithm when considering there is no error to the pursuer's initial states. The next question is by how much does the presence of error could effect the result of the optimisation algorithm.

As to know this, we consider two pursuer states, i.e. initial altitude and initial speed. The initial altitude are corrupted by using Eq. 4.

$$z_e = z_{actual} + z_{error} \tag{4}$$

where z_e is the pursuer's altitude as seen by the evader, z_{actual} is the actual pursuer's altitude and z_{error} is the introduced error.

It is assumed that the pursuer have a fixed throttle setting and its guidance system is known *a priori* by the evader.

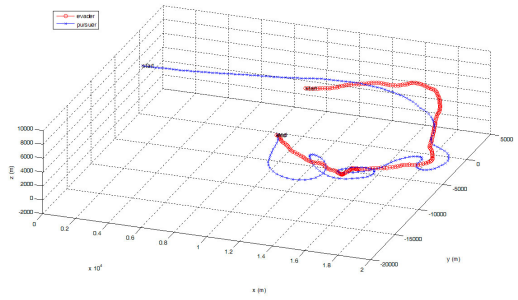

Fig. 3: The encounter in three dimension. The duration is 200 seconds.

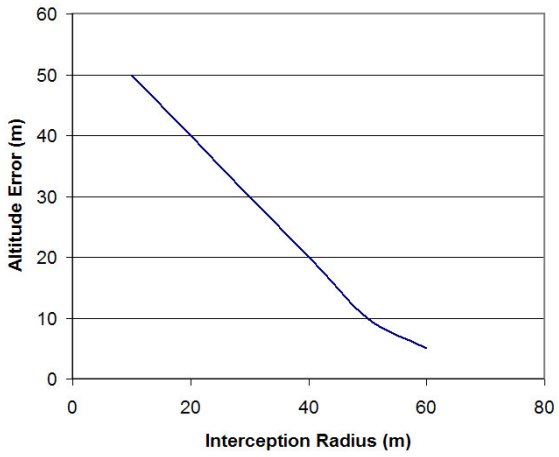

Fig. 4: Maximum error before the optimisation fail to give a good solution. Pursuer's maximum thrust is 160,000 kN.

4. Simulation Analysis

The simulation is in full three dimensions. The evader reads the pursuer's states at time, $t = 0$ s and using that information the evader optimises its trajectory against the pursuer. Example of the game is as show in Figure 3.

A number of simulations were carried out to determine the effect of error to the initial altitude of the purser as seen by the evader. The initial values are important because they are used to optimal solution against the pursuer.

When the interception radius is small, i.e. between 10 to 30 meters, the presence of errors up to $\pm$ 100 meters do not effect the outcome of the optimisation. Good solutions were still able to be found. But as the interception radius increases, the optimisation becomes more sensitive to errors. This can be seen from Figure 4.

In figure 4, when the interception radius is small, i.e. 10 meters, good solutions are still found even if the error is close to 50%. As the interception radius increases, the optimisation becomes more sensitive to initial value error. As the interception radius reaches 60 meters, even 1% error could fail the optimisation algorithm. The overall trend is the minimum error to get good solution steadily reducing as the interception radius increases.

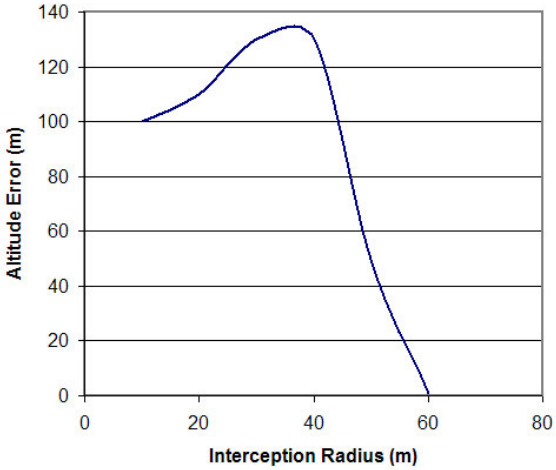

Fig. 5: Maximum error before the optimisation fail to give a good solution. Pursuer's maximum thrust is 200,000 kN.

The pursuer's maximum thrust could influence the sensitivity of the optimisation algorithm to the initial value error. Figure 5 shows the effect of errors to the optimisation algorithm with the pursuer's maximum thrust is set to 200,000 kN.

Inconsistencies between the interception radius of 20 m and 40 m is expected due to the stochastic nature of the search algorithm. The interesting part is if the pursuer's thrust is higher, the optimisation algorithm could find good solutions even if the error is relatively large. But the effectiveness is lost if the interception radius is higher than 40 m.

From Figure 5, the increase of the pursuer's maximum thrust by 60% increases the error threshold level for lower range interception radius, i.e. between 10-30 meters. This means, the optimisation could still find good solutions even if the error is more than 100 meters. However, the error threshold level significantly drop when the interception radius passes 40 meters mark. The optimisation sensitivity to errors is maximum when the interception is more than 60 meters. In this range, even a 1 meter error will fail the optimisation.

5. Conclusion

The study has shown that the initial value error could significantly reduce the accuracy of the optimisation algorithm. The magnitude of the errors are, among other, influenced by the pursuer's maximum thrust and interception radius.

The evader's optimisation algorithm could tolerate errors up to 100 meters if the pursuer's maximum thrust is high and has a small interception radius, such as 30 m. The simulation time is between 10-15 seconds which is almost real time. This opens up the possibility for its use in real time application such as UCAV.

The quality of the simulation can be improved by using six degrees-of-freedom model for both players. The use of parallel computing can improve the speed of the simulation.

This is done by distributing equally the candidates to multiple processors.

REFERENCES

[1] Isaac,R., *Differential Games*, New York: John Wiley and Sons, 1965

[2] Başar, T., Olsder G.J, *Differential Games*, New York: John Wiley and Sons, 1965

[3] Bäck,T., Fogel, D.B., Michalewicz, Z., *Evolutionary Computation 1: Basic Algorithms and Operators*, Bristol: Institute of Physics Publishing, 2000

[4] Friedman, A., *Differential Games*, New York: John Wiley and Sons, 1971

[5] Zipfel, P.H., *Modeling and Simulation of Aerospace Vehicle Dynamics*, AIAA Education Series, ed. Przemieniecki, Virginia, 2000

[6] R. Tuomas and E. Harry , *Applying Nonlinear Programming to Pursuit-Evasion Games*, System Analysis Laborator Research Reports, Helsinki University of Technology, 2000

[7] Imoda, F., "Some Practical Approaches To Pursuit-Evasion Dynamic Games", in *Cybernatics and Systems Analysis*, vol. 38, no.2, 2002, page 276-291

[8] Järmark, B., Hillberg, C., "Pursuit-Evasion Between Two Realistic Aircraft", in *Journal Guidance, Dynamics and Control*, Vol. 7, no.6, 1984, page 690-694

[9] J. Wang and S.Y. Chao, "Sensor Noise Model Development of a Longitudinal Positioning System for AVCS", in *Proceedings of the American Control Conference '99 (ACC'99; (ACC'99; June 2-4, 1999*, San Diego, CA, Session FM11-1, pp. 3760-3764.