공학석사학위논문

Development of A* Real-time Path Planning Algorithm of UAV Considering Collision Avoidance

충돌회피를 고려한 무인항공기 A* 실시간 경로생성 알고리즘 개발

2015 년 2 월

서울대학교 대학원

기계항공공학부

이 경 현

Development of A* Real-time Path Planning Algorithm of UAV Considering Collision Avoidance

충돌회피를 고려한 무인항공기
A* 실시간 경로생성 알고리즘 개발

지도교수 김 유 단

이 논문을 공학석사 학위논문으로 제출함

2015 년 2 월

서울대학교 대학원

기계항공공학부

이 경 현

이경현의 석사 학위논문을 인준함
2015 년 2 월

위 원 장 ___김___현___진____(인)

부위원장 ___김___유___단____(인)

위 원 ___김___성___완____(인)

Abstract

Unmanned aerial vehicle (UAV) has developed to perform military missions including patrol, surveillance, and reconnaissance. Recently, the applications of UAV are expanding to the private markets of UAV agriculture, aerial photography and etc. For these various applications, collision avoidance of UAV is an essential problem, because collision to obstacles may cause not only mission failure but also destruction of UAV and fatal accident including loss of human life.

In this study, modified A* path planning algorithm is proposed for UAV systems. In various robot applications, standard A* algorithm is widely used, which does not apply dynamics of UAV. Therefore, the results from the path planning algorithm should be post-processed to reflect the dynamic constraint of an UAV such as limited turning angle. To deal with this problem, search direction achievable by the UAV is considered in the proposed A* algorithm. The proposed algorithm can be implemented on the on-board system of the UAV in real-time, and does not need post processing to follow the result of path planning. Nonlinear guidance algorithm and PID controller to follow the path are also designed. The performance of the proposed algorithm is demonstrated using numerical simulations. Six degree-of-freedom simulation model was obtained by system identification flight test. Finally, the integrated algorithm is verified by the flight test.

Keywords: Unmanned Aerial Vehicle, collision avoidance, A* path planning, nonlinear guidance, real-time path planning.

Student number: 2013-20690

Table of Contents

List of Tables

List of Figures

1. Introduction

UAV has been widely used for surveillance, reconnaissance, aerial photography, and mapping, both in civil and military applications. For these various applications, stability of UAV should be guaranteed, because failure of UAV may cause not only mission failure, but also damage to structures, assets, and lives on the ground.

For the autonomous flight of the UAV, collision avoidance is an essential problem. Collision avoidance problem should consider static/dynamic obstacles, as well as threats. In this study, path-planning algorithm is considered to keep away from all threat sources.

Path planning algorithms can be categorized into graph methods, potential field methods, and grid methods[1]. Graph methods transform the path planning problem into network search problem. There are several algorithms in graph methods, which include Voronoi diagrm method generating network using Voronoi diagram[2], probabilistic road map methods[3], and rapid random tree methods[4]. One of the advantages of this approach is a balance between computing time and performance, however, it cannot be used for on-line planning[1]. Potential field method considers imaginary potential field near the obstacles, which generate a force to the UAV so that the UAV can avoid the obstacles naturally. this approach is intuitive and fast to compute, however it can be easily trapped into local optima[5]. Grid based method divides map data for path-planning, which include optimal grid based method such as Dijkstra's algorithm[6], A* algorithm that improves performance of Dijkstra's algorithm[7], and D* algorithm considering changeable map data[8].

Several methods have been applied for the path planning of UAV, by combining Voronoi diagram and potential field algorithm[2], or by using heuristic and genetic algorithm and using MILP and A* algorithm [9].

In this thesis, modified A* search algorithm is proposed for UAV, considering UAV dynamics by restricting search range.

This paper have following contributions. First, path planning algorithm considering UAV bank angle limit is proposed. The proposed algorithm does not need post processing. Second, path planning can be performed on embedded computer for real-time operation. Third, algorithm is verified by flight tests.

This thesis is organized as follow. Introduction is started in chapter 1. In chapter 2, UAV guidance and control laws are briefly summarized. Path following guidance law to follow a guidance command is described. In chapter 3, path planning algorithms are described. A* path planning algorithm is analyzed and the inherent problem of the method is described. Modified A* algorithm considering the dynamic constraint of UAV is proposed. In chapter 4, result of numerical simulation results are provided using 6-axis linear model of UAV. In chapter 5, flight test results of various scenarios are provided. Finally, in chapter 6, conclusion and future research works are addressed.

2. Guidance and Control of UAV

In this chapter, guidance and control algorithm of UAV is briefly explained. Longitudinal and lateral nonlinear guidance algorithm is used with simple PID (Proportional-Integral-Derivative) controller.

2.1 Guidance of UAV for Path Following

Guidance block of UAV generates flight path angle reference command and roll angle reference command using sensor data and other upload information from ground control system. Control block of UAV converts the reference commands to actuator inputs. Block diagram of guidance and control block are shown in Fig. 2.1.

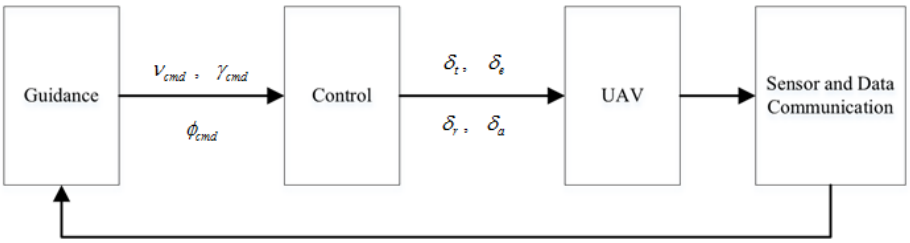


Fig. 2.1. Block Diagram of UAV System

2.1.1 Longitudinal Guidance

In this study, it is assumed that UAV missions are performed in 2D (2 Dimension), therefore longitudinal guidance is only needed to maintain height and velocity of the UAV. Reference velocity value is uploaded from the ground control system. Velocity reference of the guidance block is constant. Guidance block generates flight path angle command to maintain the altitude as follows.

$$\gamma_{cmd} = -k_{\gamma}(h - h_{cmd}) \tag{2.1}$$

In Eq.(2.1), γ_{cmd} is a flight path angle command, k_{γ} is a guidance gain of the flight path angle, h is a height of UAV, and h_{cmd} is a target height which is uploaded from the ground control system.

2.1.2 **Lateral Guidance[10]**

Lateral guidance law generates a roll angle reference. In this study, nonlinear guidance algorithm is used to make UAV to follow the desired path. To do this, a target point, which is far away from UAV at the distance of L, is chosen. Nonlinear path-follow guidance algorithm compute μ, which is an angle between ground velocity of UAV and desired vector between UAV and target point. Then, guidance command $a_{y_{cmd}}$ can be generated using μ as follows.

$$a_{y_{cmd}} = \frac{2V^2}{L}\sin\mu \tag{2.2}$$

Where, V is a velocity of the UAV, and L is a distance between UAV and the target point, which is a design parameter. Using small L, UAV can follow the desired path more closely. However, there may exist overshoot or interference. Otherwise, if large L is chosen, they UAV converges to the path slowly. Therefore, L should be selected carefully considering the type of UAV. In this study, L is tuned to proper value via flight test.

UAV follows the acceleration guidance command. When UAV turns with a specific bank angle, centripetal acceleration command is generated. When UAV maintains a specific altitude while turning of UAV is a coordinated turn, then following acceleration can be considered.

$$a_y = g\tan\phi \tag{2.3}$$

where, a_y is a lateral acceleration, g is a gravitational acceleration, and ϕ is a bank angle of the UAV. Fig. 2.2Fig. 2.3 show diagrams for the UAV guidance. By making the lateral acceleration a_y using the guidance acceleration $a_{y_{cmd}}$, Eq.(2.2), roll angle command ϕ_{cmd} can be obtained as follows.

$$\phi_{cmd} = \tan^{-1}\left(\frac{a_{n_{cmd}}}{g}\right) \tag{2.4}$$

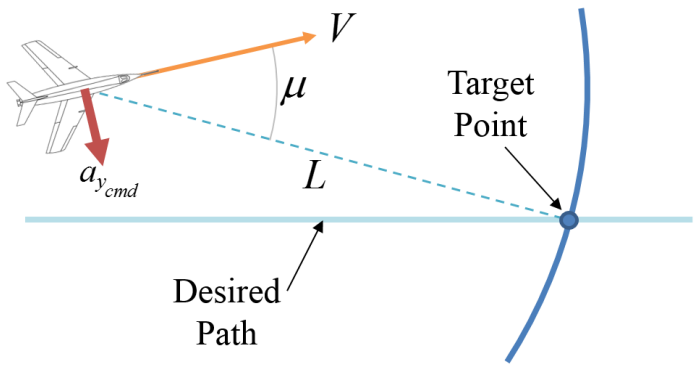
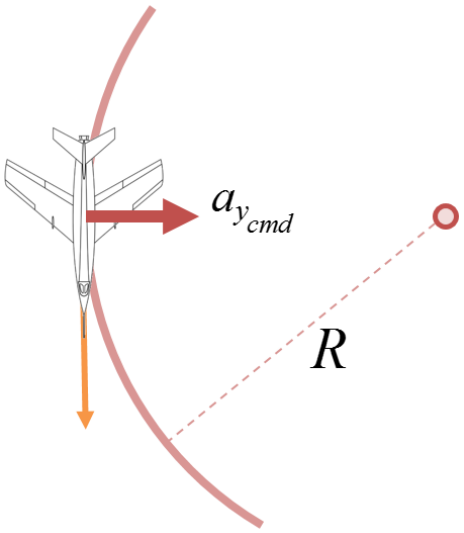


Fig. 2.2. Target Point and Lateral Acceleration of UAV



22

Fig. 2.3. Turning of UAV with Lateral Acceleration Command

In this section, guidance laws for longitudinal and lateral motion were treated. Longitudinal guidance law generates velocity and flight path angle reference commands, and lateral guidance law generates a roll angle reference command. These commands are sent to the autopilot of the UAV as inputs.

2.2 Autopilot system of UAV

2.2.1 Longitudinal Control

To make the UAV follow the commands of velocity and flight path angle, longitudinal autopilot should be designed to generate throttle and elevator actuator inputs. For Longitudinal control, angle of attack sensor is attached to UAV.

First, velocity error e_v is defined as follows.

$$e_v = v_g - v_{cmd} \tag{2.5}$$

where, v_{cmd} is as longitudinal guidance command, and v_g is a ground velocity obtained by GPS sensor.

For a speed hold, a throttle input δ_t can be generated as follows.

$$\delta_t = k_{P_t} e_v + k_{I_t} \int e_v dt \tag{2.6}$$

where $k_{P_t} > 0$, $k_{I_t} > 0$ are gains for proportional and integral feedback control.

For a pitch angle hold, pitch command θ_{cmd} can be generated using the angle of attack sensor and flight path angle reference command γ_{cmd} ,Eq.(2.1), as follows.

$$\theta_{cmd} = \theta - (\alpha + \gamma_{cmd}) \tag{2.7}$$

Pitch error e_g is defined as follows.

$$e_\theta = \theta - \theta_{cmd} \tag{2.8}$$

Then elevator input δ_e can be generated as follows.

$$\delta_e = k_{P_e} e_\theta + k_{I_e} \int e_\theta dt + k_{D_e} q \tag{2.9}$$

where $k_{P_e} > 0$, $k_{I_e} > 0$, $k_{D_e} > 0$ are gains of proportional, integral, and derivative feedback control.

2.2.2 **Lateral Control**

Lateral controller generates aileron and rudder commands using the roll angle reference command of the lateral guidance law.

Roll angle error e_ϕ is defined as follows

$$e_\phi = \phi - \phi_{cmd} \tag{2.10}$$

where ϕ_{cmd} is a roll angle guidance command, Eq.(2.4), and ϕ is a roll angle of the UAV.

For a bank hold, aileron input can be generated as follows,

$$\delta_a = k_{P_a} e_\phi + k_{I_a} \int e_\phi dt + k_{D_a} p \tag{2.11}$$

where p is a roll angular velocity, and $k_{P_a} > 0$, $k_{I_a} > 0$, $k_{D_a} > 0$ are gains of proportional, integral, and derivative feedback control.

Rudder input is generated for coordinated turn, i.e., i $\beta = 0$, as follows.

$$\delta_r = -k_{P_r}\beta \tag{2.12}$$

where sideslip angle β is obtained using the sideslip angle sensor, and $k_{P_r} > 0$ is a gain of the proportional feedback control.

In this section, UAV autopilots for longitudinal and lateral motions were treated. Autopilots convert roll angle reference command ϕ_{cmd} and flight path angle reference command γ_{cmd} to actuator inputs of aileron, elevator, throttle, and rudder. These inputs are finally converted to PWM motor commands for each motors of control surfaces and motor by flight control computer.

3. Path Planning of UAV

UAV path planning algorithm is presented in this chapter. A conventional A* path planning algorithm, which is widely used in path planning including unmanned robots is briefly summarized, and it is modified for UAV application. Numerical simulation is performed to demonstrate the performance of the proposed path planning scheme.

3.1 A* Path Planning Algorithm

For robot systems, path planning problem has been studied by several approaches. Path planning algorithms are divided into optimal approaches providing optimal solutions and real-time approaches providing real-time sub-optimal solutions. Optimal path planning algorithm generates a best solution, however, it takes a lot of computing time. Real-time path planning algorithms does not provide the best solution, but it consumes less computing time for real time applications.

Optimal path planning algorithms usually require very high computing power, which leads to extremely bulky system. It is hard for smaller unmanned robots, however, to have high computing power. Especially, UAVs have limited capacity of space and weight. Note that a mission radius of the UAV is directly related to the weight, they prefer the ways to reduce total weight. For those reasons, this paper used real-time path planning algorithm, instead of optimal algorithms.

There are two different methods in real-time path planning algorithms. First one is grid method, which divides mission area into smaller discrete areas and performs path-planning. The other one uses coordinated system directly without dividing the area.

A* algorithm is one of the grid based path planning algorithms. The A* algorithm classifies searched area and unsearched area by dividing mission area. Peter Hart, Nils Nilsson, Bertam Raphael of Stanford University suggested the algorithm in 1968, which uses heuristic method to improve time performance. It is known as best-first search algorithm, by combining the advantages of both optimal and heuristic algorithms, through fusion of the past path-cost function and a future path-cost function.

A* path planning algorithm guarantees a solution when it exists. It is extension of Dijkstra's optimal path planning algorithm, and it is widely used for path-planning because it can be done in real-time, and provides quite fine solution.

To understand A* algorithm, let us define several technical terms. "The Cost function" defines the cost value from initial position to the present position. In general, it represents moving distance of the unmanned robots, but it is not mandatory to use the moving distance. For example, it can be defined as fuel consumption, according to the problem to solve. "A Heuristic function" defines a future estimated cost form the present position to goal position. Generally, it simply estimate future cost, not considering obstacles. The function should be well-defined, because the performance of the A* algorithm depends on the design of this function. "The Open set" is a set of candidate for the next search. Area in the open set is unsearched area, and it can be searched anytime in the future. "The Closed set" is a set of searched area. It is already reflected in the cost function.

A* path finding algorithm is performed in 3 stages as shown in Fig. 3.1.

The first step is initialization stage. In the stage, the algorithm sets initial position and initial value of two cost functions, cost function and heuristic function. After the initialization stage, the open set includes one candidate position which is the initial position.

The second stage is a loop stage, which is the main stage of the A* algorithm. Algorithm iteratively executes this stage to find path to goal the

position. The beginning of the loop stage is to select one candidate in the open set, which has lowest cost function, sum of two cost functions. That is, the position which is estimated to have the lowest cost will be selected to the next search position. When the algorithm just enters the loop stage after the initialization stage, it only has one candidate, initial position, and therefore the algorithm starts it naturally from the initial position. After choosing the next search point, algorithm inserts the position in the closed set and calculates the cost to that position. Then, algorithm investigates adjacent areas to choose search positions, compute the cost value. Through these steps, the open set includes next candidate search positions. After that, algorithm goes back to the top of the loop stage, and iterate all of the steps until the chosen position from the open set is the goal position or there does not exist any position in the open set. When the goal position is chosen, then path from the initial position to the goal position is generated through the back tracking stage.

The back tracking stage is a process to determine waypoints from the initial point to the goal point. In the loop stage, when algorithm inserts next positions into the open set, it also has the information of the parent position which it comes from. With this information, algorithm can track the whole waypoints form the goal position to the initial position.

Through the whole stages, the set of waypoints is determined, and finally the determined waypoint set is the result of the path-planning.

A* algorithm should be designed considering several issues. One is which positions will be searched in the next step. The other one is how to calculate the cost and heuristic functions. In general, the next positions are selected as 8 adjacent positions in the grid, and the cost function is simple travel distance from the initial position.

In this study, the A* algorithm is modified such that search directions are limited among 8 adjacent positions. Limited search directions can be obtained considering the heading angle constraints, as shown in Fig. 3.2., which will generate path preventing extreme turning of the UAV.

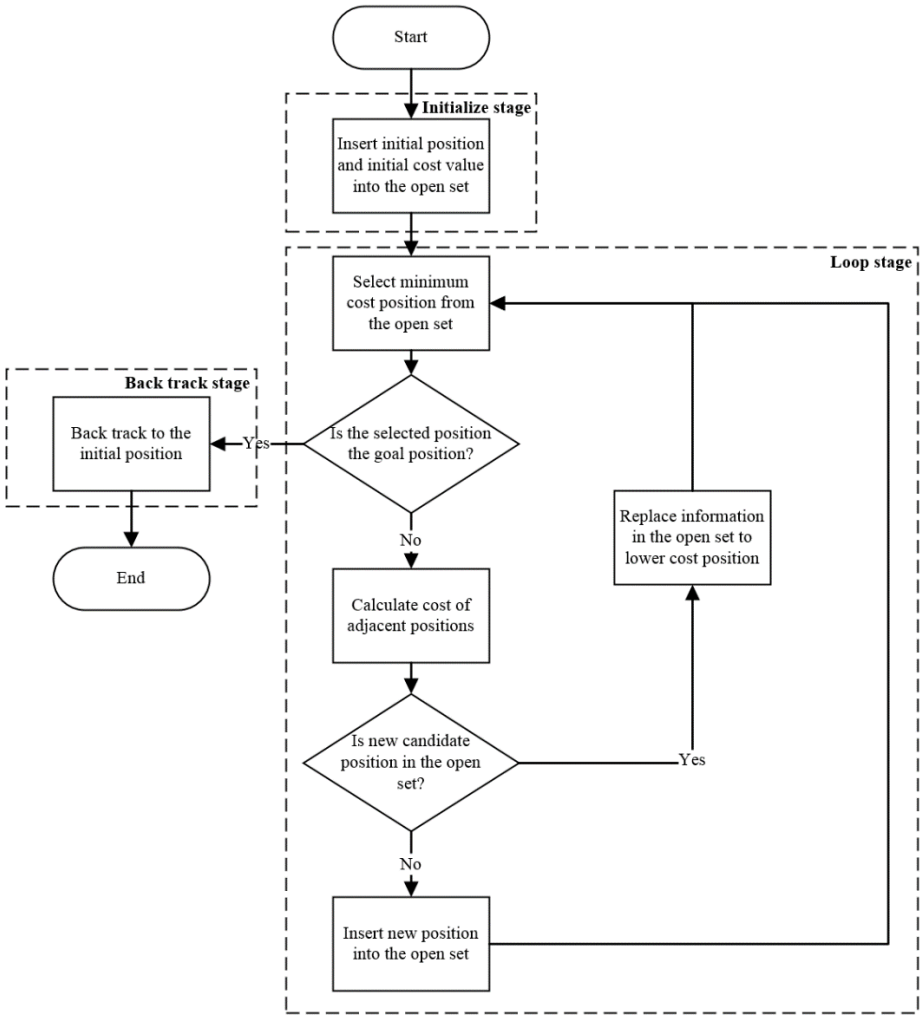


Fig. 3.1. Flow Chart of A* Path Finding Algorithm

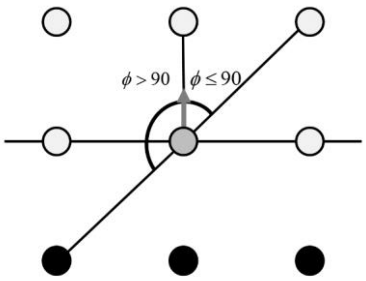


Fig. 3.2. Heading angle Constraint

3.2 Modified A* Path Planning Algorithm for UAV

In this section, modification of the A* search algorithm considering the dynamics of UAV will be described. Note that it is not easy for the UAV to change the heading angle compared to the ground robots. Also, for UAV cost function using distance only is somewhat improper for real energy consumption. In this study, a modified A* algorithm is proposed by combining the A* algorithm and a simple kinematics of the UAV.

3.2.1 Problems of Standard A* Search

To select the next search position is the hardest part for the UAV in standard A* algorithm. In general, A* searches 8 positions adjacent to the current position, but sometimes the fixed-wing UAV cannot follow the designed path because of the dynamics. Fig. 3.3 shows a sample result of the standard A* algorithm. The area marked by small squares are the obstacles. The A* algorithm successfully generated a path from an initial position to goal position without collision with the obstacles, however there exists a steep turning part. Fig. 3.4. shows the result of post processing considering the dynamics of UAV.

In this study, the A* algorithm is modified to generate proper path for the UAV by considering UAV search direction in path finding stage.

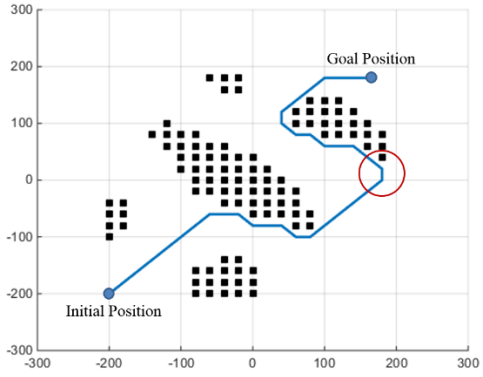


Fig. 3.3. Standard A* Search Result and Impossible Point for UAV

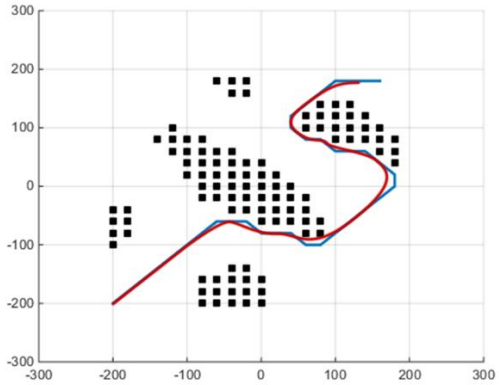


Fig. 3.4. Post Processed Standard A* Path

3.2.2 Search Candidate Point Selection

In the previous section, it is shown that the standard A* search investigating adjacent 8 points may results in improper path for the UAV because it does not consider the dynamics of the UAV.

In this Study, a turning radius of the UAV is considered in the process of point search. The merit of the method is that it can consider various turning radius of different UAVs.

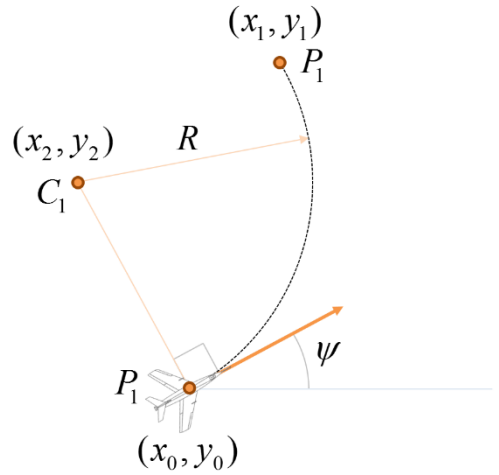


Fig. 3.5. Problem Condition Description

Consider the initial position of the UAV is (x_0, y_0), the next position to be searched is (x_1, y_1). Let us imagine a circular arc consisting of (x_0, y_0) and (x_1, y_1) centered at (x_2, y_2) as shown in Fig. 3.5, R is a radius of the circle, the circle is tangent to the velocity vector of the UAV at (x_0, y_0), and ψ is a heading angle of the UAV.

In this study, UAV can move from the present position to the next position by circle path or straight path. When UAV reached to the next point P_1, then UAV's heading is an initial heading of next search. Fig. 3.6 shows the several cases for the given initial position and velocity vector. According to the next search position, the heading angle at the position can be obtained by the geometry of the circle.

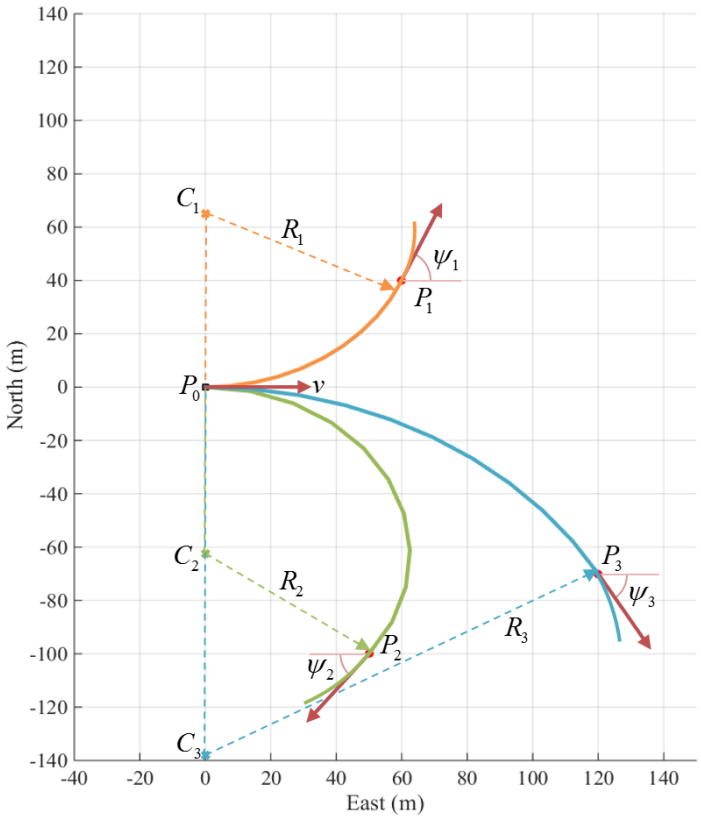


Fig. 3.6. Turning of UAV According to the Next Position

Note from Fig. 3.5 that the center point (x_2, y_2) of turning circle can be calculated using (x_0, y_0) and (x_1, y_1) with respect to ψ as follows.

Case I: $\tan(\psi) = 0$

Case: $y_1 = y_0$

Not a circle (straight path)

Case: $y_1 \neq y_0$

$$x_2 = x_0 \tag{3.1}$$

$$y_2 = \frac{(x_1 - x_0)^2}{2(y_1 - y_0)} + \frac{y_1}{2} \tag{3.2}$$

Case II: $\psi = \pm\frac{\pi}{2}$

Case: $x_1 = x_0$

Not a circle (straight path)

Case: $x_1 \neq x_0$

$$x_2 = \frac{x_1 - x_0}{2} + \frac{(y_1 - y_0)^2}{2(x_1 - x_0)} \tag{3.3}$$

$$y_2 = y_0 \tag{3.4}$$

Case III: $\tan(\psi) \neq 0$, $\psi \neq \pm\frac{\pi}{2}$

Case: $y_1 - y_0 = (x_1 - x_0) \cdot \tan(\psi)$

Not a circle (straight path)

Case: $y_1 = y_0$

$$x_2 = \frac{x_1 - x_0}{2} \tag{3.5}$$

$$y_2 = -\frac{1}{\tan(\psi)} x_2 \tag{3.6}$$

Case: $y_1 - y_0 \neq (x_1 - x_0) \cdot \tan(\psi)$

$$x_2 = \frac{\frac{(x_1 - x_0)^2}{2y_1} + \frac{(y_1 - y_0)}{2}}{\frac{(x_1 - x_0)}{(y_1 - y_0)} - \frac{1}{\tan(\psi)}} + x_0 \tag{3.7}$$

$$y_2 = -\frac{1}{\tan(\psi)}(x_2 - x_0) + y_0 \tag{3.8}$$

The grid size n is determined as follows.

$$n = \frac{R_l}{D_g},\ R_l > D_g \tag{3.9}$$

where R_l is a turning radius, and D_g is a grid length.

The next position in $\pm n$ gird positions in both axis will be searched. The turning radius to those positions will be calculated, and the positions having larger turning radius compared to minimum turning radius R_m will be chosen.

By this process, UAV can choose next candidate positions according to the heading angle. If the algorithm computes this in every searches, it loses the merit of A* search algorithm because it needs lots of computation. On the other hand, let us focus on the feature that D and R_m are constant from the beginning of the search, and small changes on the heading angle does not significantly change the next search positions. Therefore, the next search positions can be pre-calculated using some range of heading angles.

In this study, total 24 ranges of searching direction is used by dividing the circle into 15 degree. Let us consider a following example for the position selecting algorithm.

Assume that UAV's initial position is origin as shown in Fig. 3.7. R_l is set to $60m$, D_g is set to $20m$, and n is set to 3. Fig. 3.7 shows area and the next search positions that UAV can reach considering its turning radius.

More precision search result can be obtained by choosing small value of D_g, but number of the next search positions will be increased.

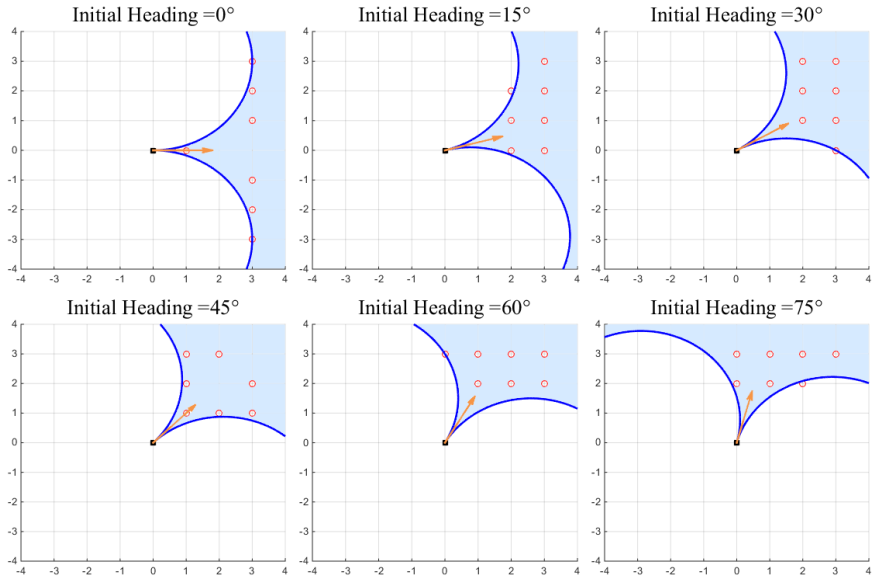


Fig. 3.7. Reachable Areas and the Next Search Position In Various Heading Range

3.2.3 **Cost and Heuristic Function Design**

The cost function and heuristic function are very important in A* algorithm, because A* determines next search position that has minimum value, i.e., sum of these two functions. And the performance of the algorithm is directly affected by the search order. In this section, design process of the cost and heuristic functions will be discussed considering UAV dynamics.

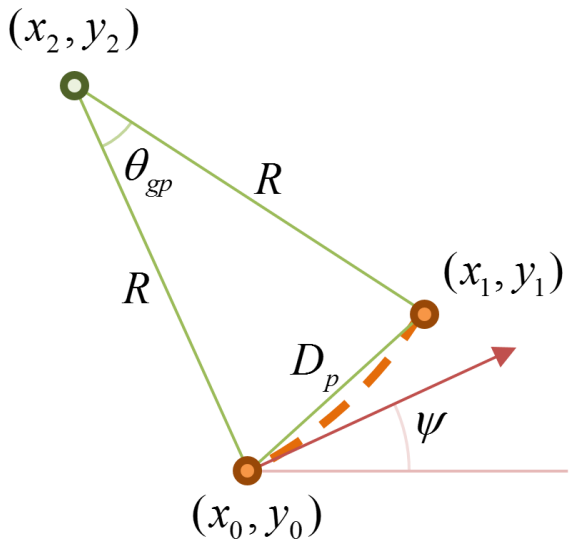


Fig. 3.8. Moving Distance of UAV on a Circular Path

Generally, the cost function is defined as a distance in robot applications. In the case of ground robot systems, almost all of the energy is consumed in the wheel motors, and therefore the cost function defined by considering the moving distance is fairly reasonable. In the case of UAV, however, energy consumption is not only in the engine motors, but also from drag caused by attitude change. In another words, even though the flying distance of two different path is same, there may be difference in energy consumption because of the path quality. Note that the energy consumption is minimized when UAV keeps its attitude same, because control surfaces used to change the attitude also consume the energy.

To reflect this phenomenon, in this study, a change of attitude of the UAV is considered in the cost function. If UAV changes its attitude, then the cost is increased. The cost function $g(x)$ is defined as follows.

$$g(x) = R \cdot \theta_{gp} + k_{\phi} \cdot \Delta\phi_g \tag{3.10}$$

where $R \cdot \theta_{gp}$ is the moving distance as shown in Fig. 3.8, $\Delta\phi_g$ is the attitude change ant k_ϕ is a weighting constant.

The value of θ_{gp} can be calculated as follows.

$$\theta_{gp} = \cos^{-1}\left(\frac{2R^2 - D_p{}^2}{2R^2}\right) \tag{3.11}$$

where R is the radius of the turning circle from the previous section, θ_{gp} is the angle of the arc to the next position. The angle $\Delta\phi_g$ is roll angle change of UAV for turning, which is the difference of between the roll angle of the current position and that of the next position. The angle $\Delta\phi_g$ can be calculated considering the turning radius R and a cruise velocity V as follows.

$$\phi = \tan^{-1}\left(\frac{V^2 / R}{\mathrm{g}}\right) \tag{3.12}$$

Note that Eq.(3.12) is derived by assuming that UAV performs a coordinated turn.

Now, let us look at the cost function, Eq. (3.10), which consist of two parts; one is a distance term $R \cdot \theta_{gp}$ that UAV actually moves, and the other one is the modified term $k_\phi \cdot \Delta\phi_g$ for the attitude change.

Similarly, the heuristic function $h(x)$ can be defined as the combination of a distance of path from the next position to the goal position as follows.

$$h(x) = R \cdot \theta_{hp} + k_\phi \cdot \Delta\phi_h \tag{3.13}$$

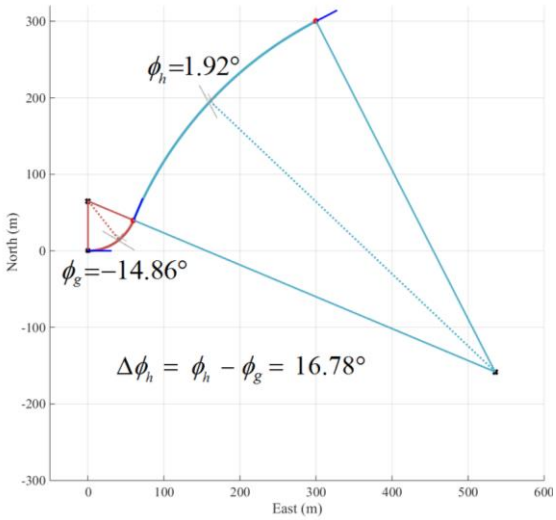


Fig. 3.9. $h(x)$ **Calculation and Roll Angle Change through a Path**

Fig. 3.9 shows roll angle change of two circular paths.

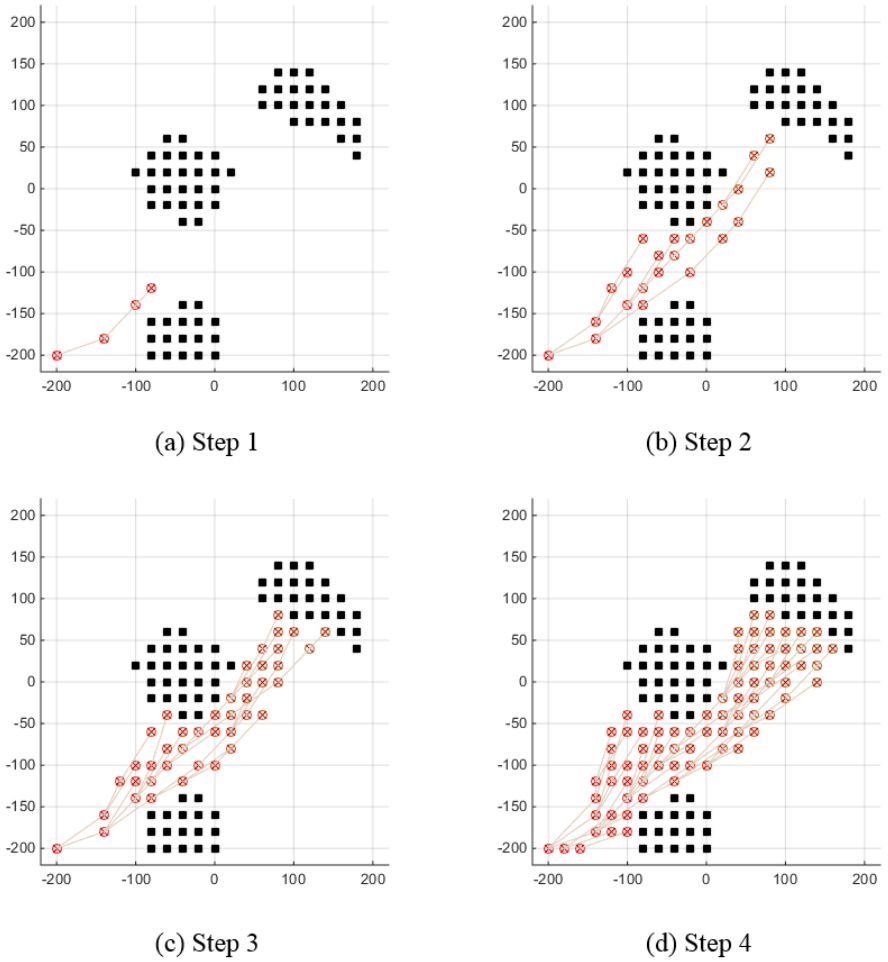
3.2.4 **Modified A* Path Planning Simulation**

In this section, simulation result using the modified A* algorithm is presented. Note that the simulation of this section does not consider the guidance and control of UAV.

Simulations are performed for same map data with different initial heading angles. Then, simulations are done using different k_ϕ to examine the influence of attitude change in the cost function.

Fig. 3.11-Fig. 3.13 show the path-planning results for different initial heading angles; $0°$, $45°$, and $90°$. When the initial heading angle is changed to $45°$ or $90°$ as shown in Fig. 3.12 and Fig. 3.13 , UAV turns north to find a path. Note that straight path has the best efficiency among every paths, UAV tries to search straight path if possible. However when the UAV encounters obstacle, the path should be turned.

Fig. 3.10 shows the search process of the modified A* algorithm for zero heading angle case, $\psi = 0°$.



(a) Step 1 (b) Step 2

(c) Step 3 (d) Step 4

Fig. 3.10 Search Process (heading angle: $0°$)

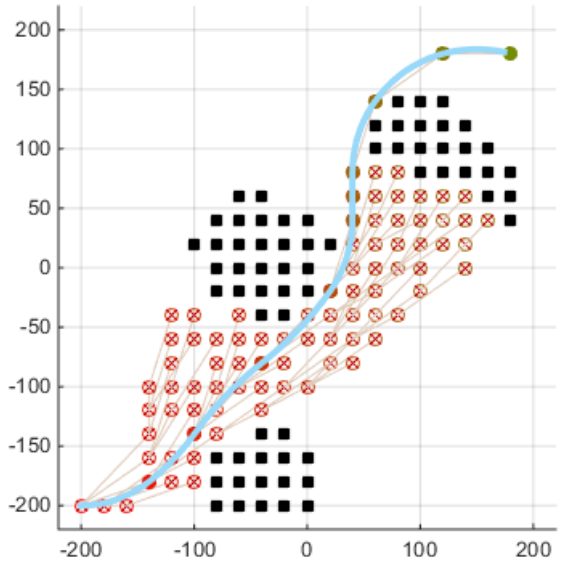


Fig. 3.11. Simulation with Initial Heading $0°$ ($k_\phi = 50$)

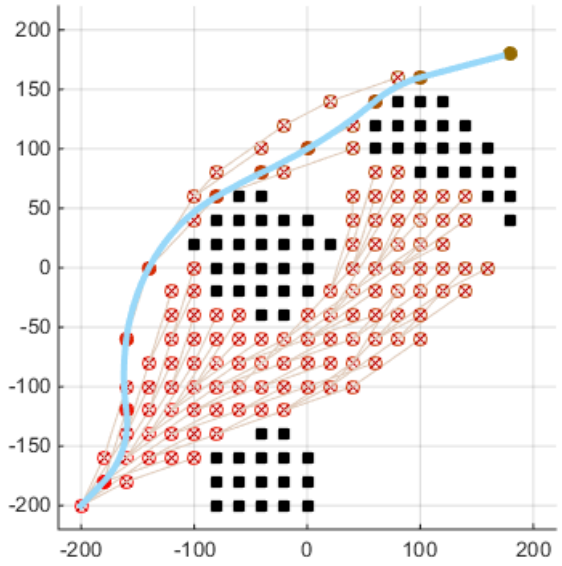


Fig. 3.12. Simulation with Initial Heading $45°$ ($k_\phi = 50$)

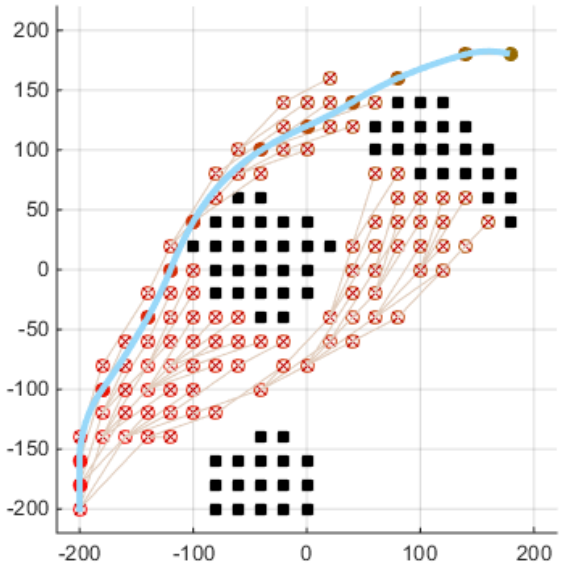


Fig. 3.13. Simulation with Initial Heading $90°$ ($k_\phi = 50$)

In the simulation results of Fig. 3.11-Fig. 3.13, k_ϕ is set as 50, which means that $1\,rad$ change of the roll angle is same as the $50\,m$ distance moving. Fig. 3.14 shows the simulation result for $k_\phi = 100$, where the attitude change is more weighted than $k_\phi = 50$ case. Compared to Fig. 3.11 ($k_\phi = 50$), the generated path is more smooth as shown in Fig. 3.14 ($k_\phi = 100$). As expected, UAV does not change the roll attitude much.

All the above simulations are performed based on the Matlab code. As summarized in Table 3.1, the average simulation time is $0.3829s$. It can be stated that the path-planning algorithm can be performed in real time.

In this chapter, standard A* path planning algorithm was introduced, and the method was modified for the application of UAV. Compared to the standard A* algorithm searching adjacent 8 positions, modified algorithm searches limited range considering the UAV turning radius. The standard A* algorithm

calculates the cost function using the moving distance only, but the attitude change is also considered in the cost function of the modified algorithm.

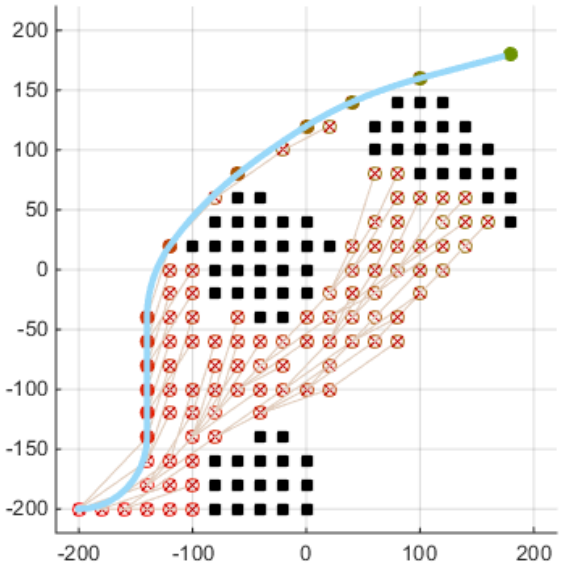


Fig. 3.14. Simulation with $k_{\phi} = 100$ ($\psi = 0°$)

Table 3.1. Calculation Time for Various Simulations

Simulation Factor	Time (s)
0°	0.4718
45°	0.4216
90°	0.3813
50	0.3044
150	0.3354
Average	0.3829

4. Numerical Simulation

In this chapter, to demonstrate the performance of the proposed path planning algorithm considering the UAV dynamics, simulation are provided. Model of UAV obtained from system identification flight test.

4.1 UAV System Modeling

4.1.1 UAV Model

Inappropriate guidance and control commands will threat the safety of UAV. Therefore, thorough verification of the algorithm before flight test is essential. For this, verification of the guidance and control algorithm by numerical simulation considering dynamics of UAV is required. Linear time-invariant system model of the UAV is obtained by system identification (ID).

The considered UAV is “Skyscout” from Hitec ltd. The body of the aircraft is composed of EPO, and motor connections and control surfaces are adequate for various missions.

4.1.2 UAV System ID

In the ID process, pre-defined command as summarized in Table 4.1 are engaged to the aircraft. Responses of the aircraft are transferred to the ground control system. Linear UAV model is calculated using the obtained flight data.

Table 4.1. Input Command of System ID Flight

	Input	Time	Change amount
Aileron	3-2-1-1	0.25s	5°
Elevator	3-2-1-1	0.5s	5°
Throttle	Doublet	2s	300gf
Rudder	3-2-1-1	0.25s	5°

Flight data is recorded in 50Hz, the linear model is obtained by minimizing the error between simulation result and flight test result. The longitudinal and lateral system is formulated as follows.

Longitudinal:

$$\begin{bmatrix} \dot{V} \\ \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & 0 & a_{24} \\ 0 & 0 & 0 & 1 \\ a_{41} & a_{42} & 0 & a_{44} \end{bmatrix} \begin{bmatrix} V \\ \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ 0 & 0 \\ b_{41} & b_{42} \end{bmatrix} \begin{bmatrix} \delta_t \\ \delta_e \end{bmatrix} \tag{4.1}$$

or,

$$\dot{\mathbf{x}}_{long} = \mathbf{A}_{long}\mathbf{x}_{long} + \mathbf{B}_{long}\mathbf{u}_{long} \tag{4.2}$$

Lateral:

$$\begin{bmatrix} \dot{\beta} \\ \dot{\phi} \\ \dot{p} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & 0 & 1 & a_{24} \\ a_{31} & 0 & a_{33} & a_{34} \\ a_{41} & 0 & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} \beta \\ \phi \\ p \\ r \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ 0 & 0 \\ b_{31} & b_{32} \\ b_{41} & b_{42} \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \tag{4.3}$$

or,

$$\dot{\mathbf{x}}_{lat} = \mathbf{A}_{lat}\mathbf{x}_{lat} + \mathbf{B}_{lat}\mathbf{x}_{lat} \tag{4.4}$$

4.1.3 **Longitudinal UAV System Model**

Longitudinal linear model is identified as follows.

$$A_{long} = \begin{bmatrix} 0 & 25.0414 & -9.7545 & 0 \\ -0.4130 & -12.0086 & 0.9435 & 2.2203 \\ 0 & 0 & 0 & 1 \\ -0.1204 & -23.5155 & 0 & -0.6636 \end{bmatrix} \tag{4.5}$$

$$B_{long} = \begin{bmatrix} 0.0080 & 10.0711 \\ 0.0003 & 3.2113 \\ 0 & 0 \\ -0.0039 & -15.3344 \end{bmatrix} \tag{4.6}$$

4.1.4 **Lateral UAV System Model**

Lateral linear model is identified as follows.

$$A_{lat} = \begin{bmatrix} 0 & 0.7231 & -0.1718 & -1.6319 \\ 0 & 0 & 0.9957 & -0.0942 \\ -53.0053 & 0 & -10.5207 & 10.4250 \\ 7.3627 & 0 & -3.3925 & -4.8898 \end{bmatrix} \tag{4.7}$$

$$B_{lat} = \begin{bmatrix} -1.4129 & -1.7054 \\ 0 & 0 \\ -51.1925 & 8.2723 \\ -9.0546 & -13.6097 \end{bmatrix} \tag{4.8}$$

4.2 Linear Model Based A* Simulation

4.2.1 **Simulink Simulator Block**

Simulink of Mathwork Ltd. is used for the simulation of the modified A* algorithm. The developed simulator consist of guidance blocks, control blocks, and linear model blocks, as shown in Fig. 4.1.

First, guidance and control blocks read the upload data of the ground control system to configure the given mission, and then control values for each control surfaces are calculated using the values of sensor.

Sensor block generates the sensor value using the state values in the linear model. Guidance blocks calculates the guidance command and the control block computes control input to follow the guidance command using the sensor value.

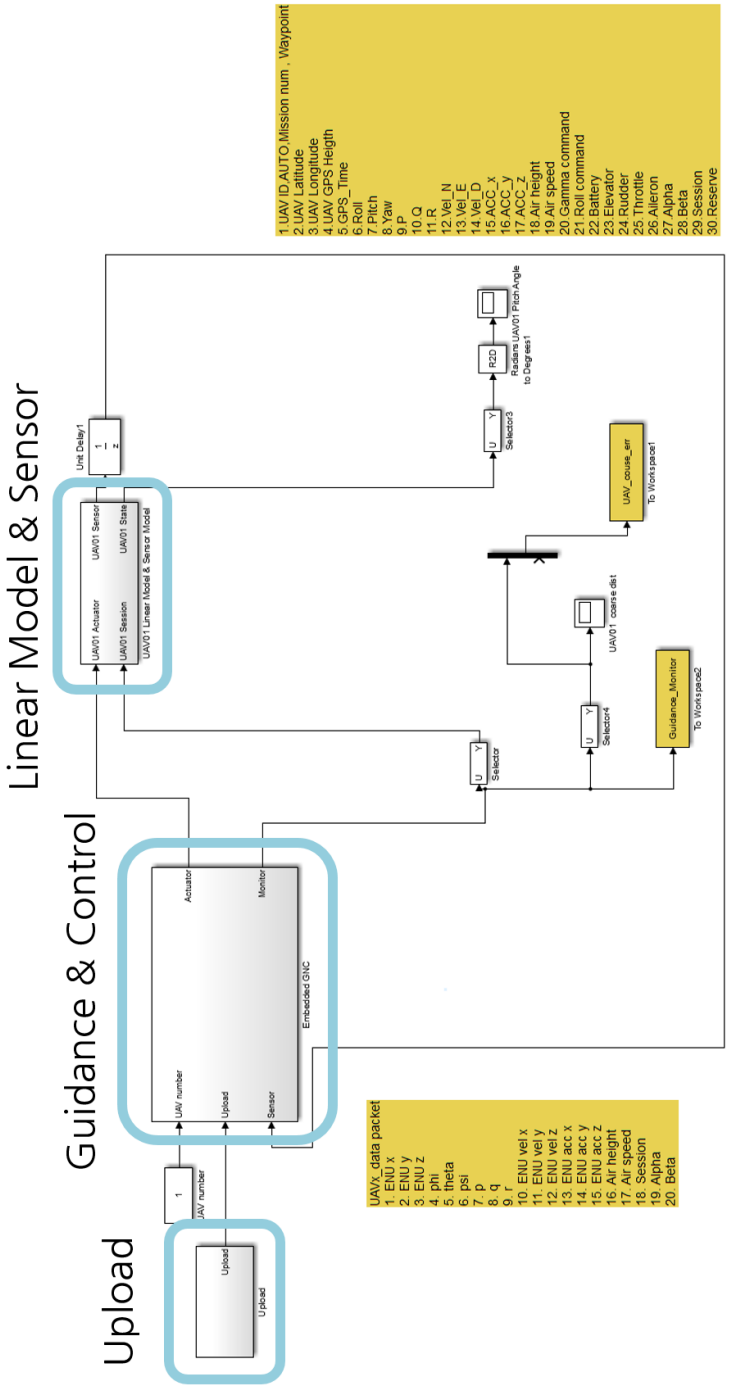


Fig. 4.1. Block Diagram of Simulator

Data from the ground control system have following packet form.

Header is a data that indicates the beginning of the data, it has “1,000” value.

Mission is a number of the assigned mission. For example, 0 is for manual flight, 1 is for loitering flight, etc. Number 1-3 is for the verification of flight quality, and 4 is for the test of the modified A* algorithm. In the simulator, a simulation can test only one mission for each simulation case, because the constant mission value is set in the simulator. In the real flight test, the number can be changed in real-time so that the ground control system can upload the number and perform any mission at any time.

Guidance L is a constant used in the guidance block, which is used to test guidance performance for different R. It is fixed to 50 by analyzing several flight test results.

Rudder trim is a correction constant for small rudder error.

Target Height is the height that UAV holds when it performs automatic missions. UAV’s longitudinal guidance algorithm generates a command to maintain this target value.

Target phi is a constant for flight performance verification mission, especially for control performance test controls UAV to follow this roll angle.

Loiter R is a constant for flight performance verification mission, for guidance performance test. R is a radius of loitering circle centered at origin point.

Zero LLH means Longitude, Latitude, Height value of the origin point. When this values are set as $(0,0,0)$, then it converts LLH values from GPS sensor to ENU coordinate system. This values are meaningless in the numerical simulation, zero LLH value should be set to the specific value near the flight test site.

Beta gain is a constant for obtaining better turning performance. It is a correction gain for the sideslip when UAV is turning with sideslip angle sensor on the UAV.

Cruise Vel is a cruise velocity of UAV. Constants, Vel Kp, Vel Ki are PID gains for maintaining the velocity. In the flight test, this value is set to $13m/s$.

Other variables are reserved variables to match the data length with UAV data packet. Because the ground control system acts like a UAV in the flight test, it is very easy to implement to the ground control system and UAV of the upload data packet has same data length with UAV data packet. Table 4.2 summarizes data packets of upload data, and Fig. 4.2 shows Simulink simulation blocks for upload data.

Table 4.2. Data Packet of Upload Data

Byte	1-4	5-8	9-12	13-16	17-20
Data	Header	Mission	Guidance L	Rudder Trim	Target Height
21-24	25-28	29-32	33-36	37-40	41-44
Target Phi	Loiter R	zero LLH_0	zero LLH_1	zero LLH_2	Beta Gain
45-48	49-52	53-56	57-60	61-120	
Cruise Vel	Vel Kp	Vel Ki	Formation K	Resv	

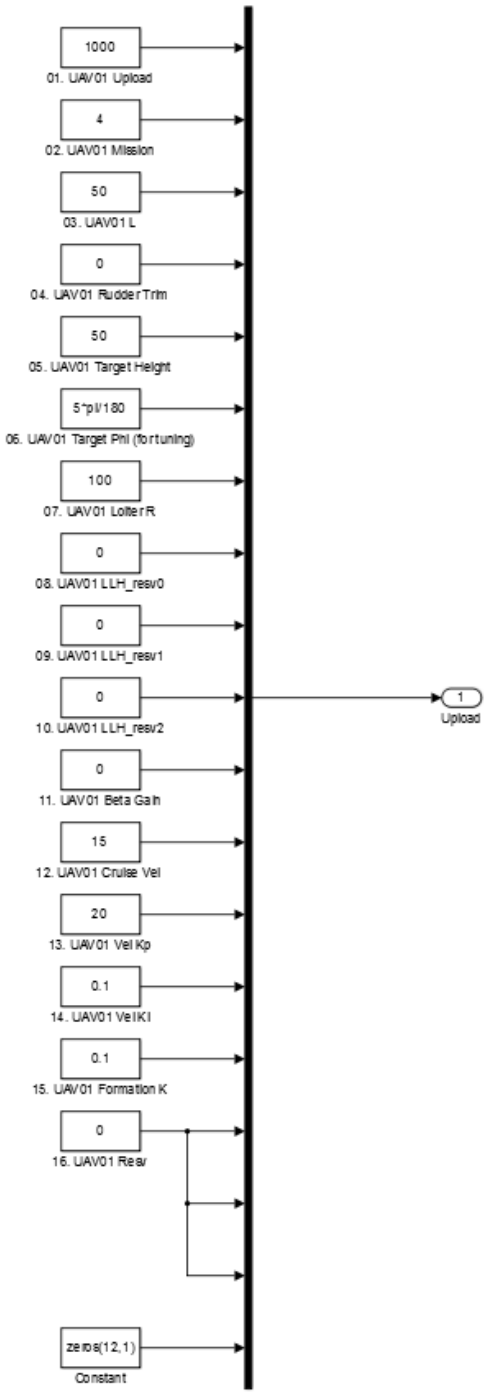


Fig. 4.2. Simulation Blocks for Upload

Upload variables are sent “Embedded GNC” block. This block consists of the guidance and control blocks as shown in Fig. 4.3. The guidance block computes roll angle reference and flight path angle reference using sensor and upload information. The computed commands are sent to the control blocks, where the commands actuator inputs are calculated using the guidance command.

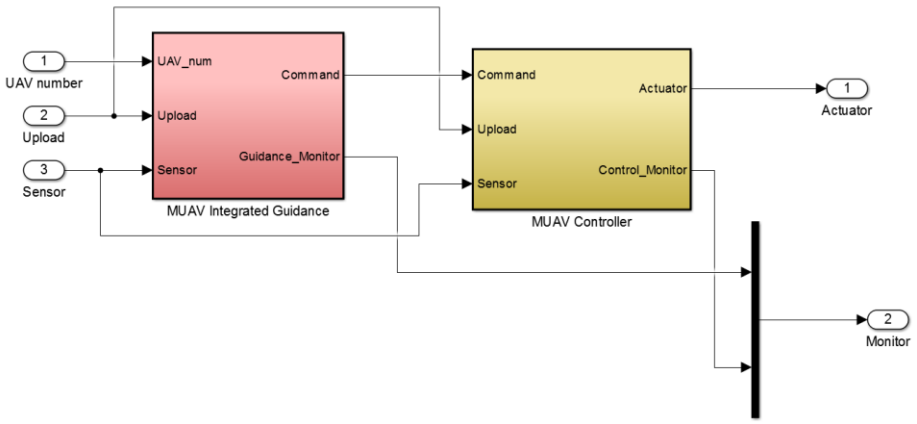


Fig. 4.3. Guidance and Control Blocks in Embedded GNC Block

In this study, the guidance block use nonlinear guidance law as described in Chapter 2. First, target position is calculated from the UAV position and desired path, and roll angle reference and flight path angle reference are generated to follow the point. This algorithm was developed for the UAV to follow any general path, and therefore it can be used to follow the generated flight path by A* algorithm. To do this, the guidance block is required to calculate the target point on the path.

Table 4.3. Information of Way Point of Modified A* Algorithm

Order	Data
1	X position of way point
2	Y position of way point
3	Arrived heading angle
4	Arrived roll angle
5	X position of center of turning circle
6	Y position of center of turning circle
7	The cost function
8	The heuristic function

4.2.2 **Angle of Sight Target Point Calculation**

The flight path from the modified A* search is given as a list of way points, as summarized in Table 4.3. Guidance block should calculate the target point on the path as follows.

Target point (x_t, y_t) is the intersection of an imagine circle which is centered at UAV's position and the path. Partial paths from the modified A* search is always arc or line. In both cases, target point (x_t, y_t) can be calculated using following equations. Note that the starting point of the partial path is $(x_0,\ y_0)$, and the end point of the partial path is $(x_1,\ y_1)$. When the path is an arc, the center of turning circle is $(x_2,\ y_2)$, and turning radius is R. A distance between UAV's position and $(x_2,\ y_2)$ is defined as D.

Linear Path:

Case I: $x_1 = x_0$

$$a = \frac{(y_1 - y_0)}{(x_1 - x_0)},\ b = -\frac{(y_1 - y_0)}{(x_1 - x_0)}(x_0 - x) + (y_0 - y) \tag{4.9}$$

$$x_{t1} = \frac{-ab + \sqrt{L^2 + a^2 - b^2}}{1 + a^2}, \quad y_{t1} = ax_{t1} + b \tag{4.10}$$

$$x_{t2} = \frac{-ab - \sqrt{L^2 + a^2 - b^2}}{1 + a^2}, \quad y_{t2} = ax_{t2} + b \tag{4.11}$$

Case II: $x_1 \neq x_0$

$$x_{t1} = x_0 + x, \;\; y_{t1} = \sqrt{L^2 - (x_0 - x)^2} + y \tag{4.12}$$

$$x_{t2} = x_0 + x, \;\; y_{t2} = -\sqrt{L^2 - (x_0 - x)^2} + y \tag{4.13}$$

Arc path:

Let us perform parallel translation and rotation transform for convenience

$$x_{21} = x_2 - x, \;\; y_{21} = y_2 - y \tag{4.14}$$

$$\theta_1 = \tan^{-1}\left(\frac{y_{21}}{x_{21}}\right) \tag{4.15}$$

$$\begin{bmatrix} x_{22} \\ y_{22} \end{bmatrix} = \begin{bmatrix} \cos(-\theta_1) & -\sin(-\theta_1) \\ \sin(-\theta_1) & \cos(-\theta_1) \end{bmatrix} \begin{bmatrix} x_{21} \\ y_{21} \end{bmatrix} \tag{4.16}$$

$$\theta_2 = \cos^{-1}\left(\frac{L^2 + D^2 - R^2}{2LD}\right) \tag{4.17}$$

$$x_{t12} = L\cos(\theta_2), \; y_{t12} = L\sin(\theta_2) \tag{4.18}$$

$$x_{t22} = L\cos\left(\theta_2\right), \; y_{t22} = -L\sin(\theta_2) \tag{4.19}$$

$$\begin{bmatrix} x_{t11} \\ y_{t11} \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1) \end{bmatrix} \begin{bmatrix} x_{t12} \\ y_{t12} \end{bmatrix} \tag{4.20}$$

$$\begin{bmatrix} x_{t21} \\ y_{t21} \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1) \end{bmatrix} \begin{bmatrix} x_{t22} \\ y_{t22} \end{bmatrix} \tag{4.21}$$

In each case, derived (x_{t1}, y_{t1}) and (x_{t2}, y_{t2}) are points of intersection on the path. Therefore let us choose a point that is close to the next way point (x_1, y_1). Using (x_t, y_t) as target point of the nonlinear guidance algorithm, the flight path angle and roll angle references can be computed.

4.2.3 **Simulation Result**

First of all, some specific missions are performed to verify the guidance and control performance.

For the case that the mission value of upload variables is 1, UAV follows roll angle reference command. For the case that the mission value is 2, UAV's guidance block generates a path to the north passing the origin. With this path, UAV flies to the north or the south according to the initial heading angle. For the case that the mission value is 3, UAV will have the path of circle which is centered at the origin with specific radius that is uploaded.

Via mission 1, the control performance can be examined; how well the controller of UAV follows the roll angle reference command. Via mission 2 and 3, the guidance performance can be examined; how well the UAV follows two basic paths, i.e., line and circle.

As shown in Fig. 4.4 and Fig. 4.5, the flight path of the UAV converges to exact circle for the mission 1. Note that there is no disturbance in the simulation. For mission 2, UAV converges to the path heading north as shown in Fig. 4.6 and Fig. 4.7. UAV firstly flies directly to the path. As the path is getting closer, it turns to the north to converge the path. For the mission 3, the path of the UAV converges to the exact circle that with $100\,m$ radius. As shown in Fig. 4.8 and Fig. 4.9.

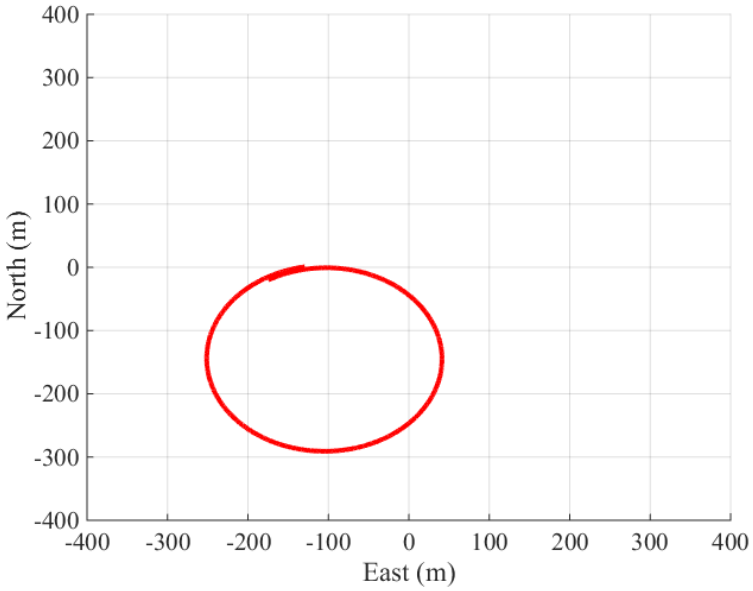


Fig. 4.4. Mission 1 Simulation Result

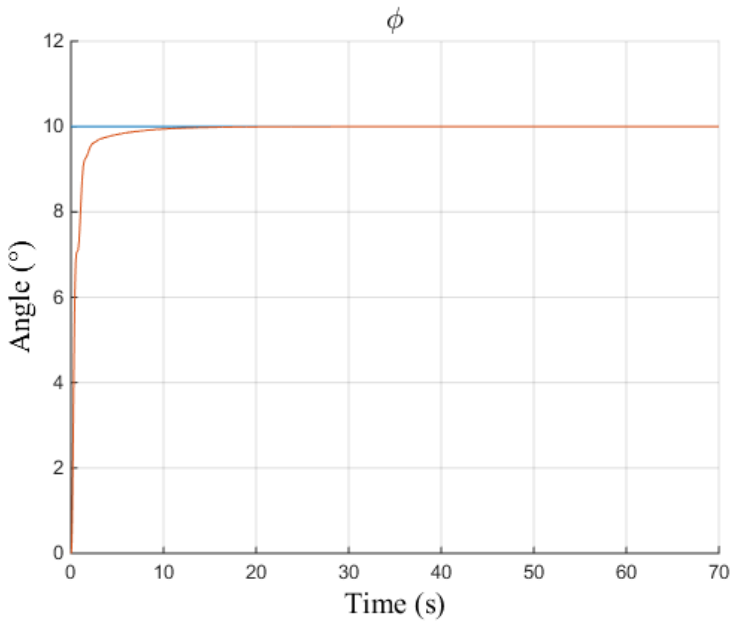


Fig. 4.5. Mission 1 Roll Angle Command and Response

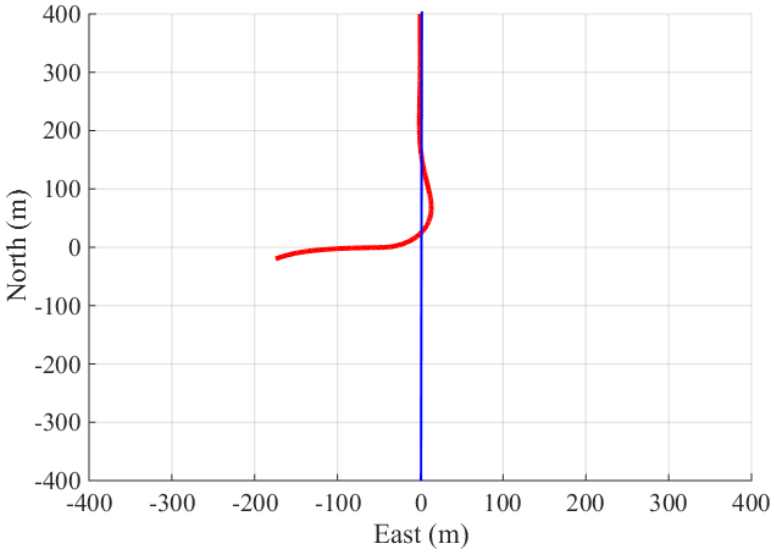


Fig. 4.6. Mission 2 Simulation Result

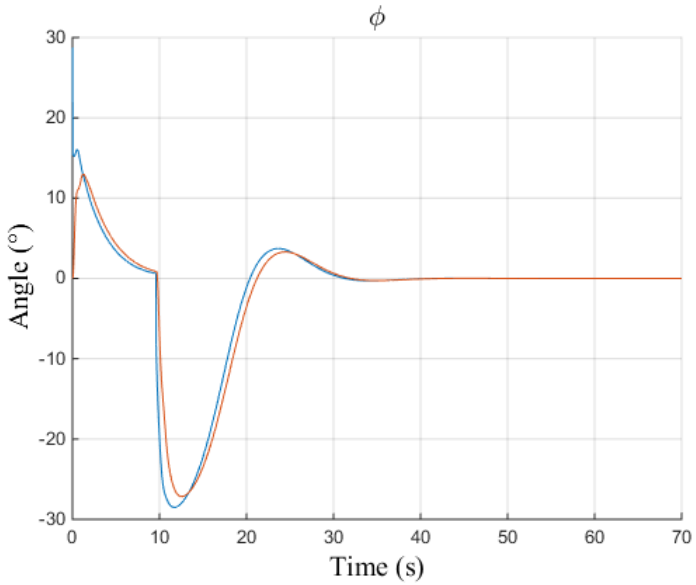


Fig. 4.7. Mission 2 Roll Angle Command and Response

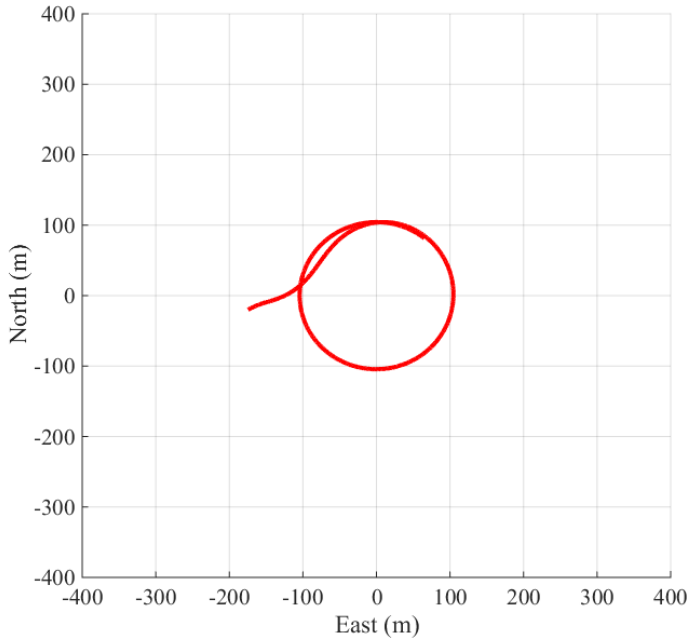


Fig. 4.8. Simulation Result of Loitering Path

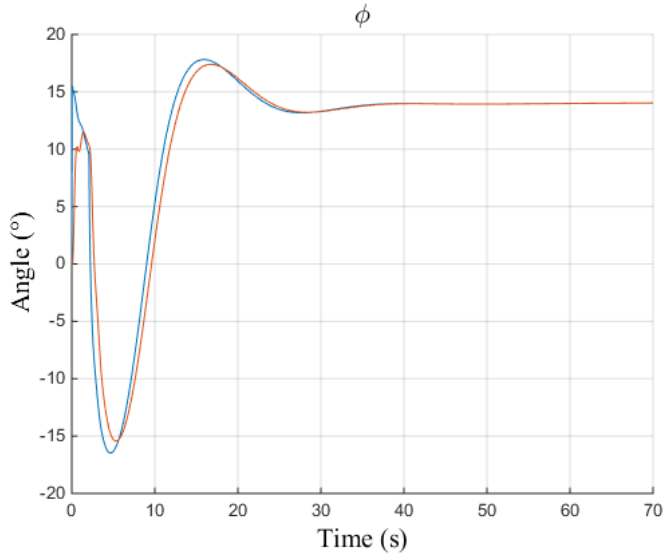


Fig. 4.9. Mission 3 Roll Angle Command and Response

For the case that the mission value of upload packet is 4, UAV starts to find path using the modified A* algorithm. Simulations are performed using real map data for flight test. Three test cases are performed, all have different initial heading angle. Two of them have same map data, and the other one has different map data. Simulation 1 and 2 use Map Data I, and the initial heading angles are $0°$ and $90°$, respectively. Simulation 3 uses Map Data II, and the initial heading angle is $30°$.

Fig. 4.10 shows standard A* search result. The generated path is composed using only line segments, UAV may not follow the path because of the limit of the turning radius. Using this map data, simulation results using the proposed method as shown in Fig. 4.11 and Fig. 4.12, for the cases that initial headings are $0°$ and $90°$, respectively. In Fig. 4.11 and Fig. 4.12, small rectangle denotes a way point, small triangle denotes a current position of the UAV, and small x letter denotes a target position at the current position. It can be stated that the target positions are properly generated.

Fig. 4.13 shows the result of standard A* using Map Data II. Again, standard A* result is not proper to UAV. Fig. 4.14Fig. 4.16 show a simulation result using the modified A* path-planning method, UAV follows the path pretty well. Map Data II is slightly more complicated compare to Map Data I, but it is not exceeding the turning radius limit of the UAV.

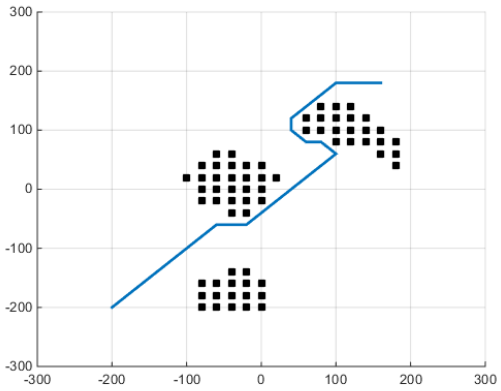


Fig. 4.10. Standard A* Search Result for Map Data 1

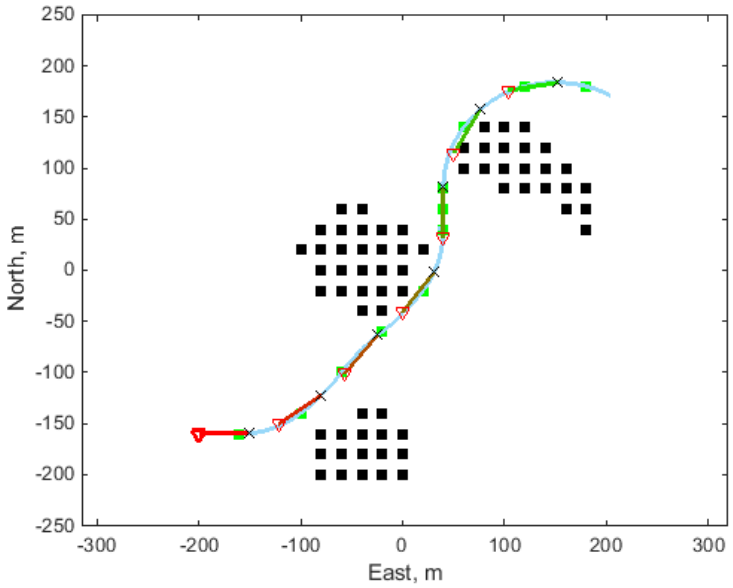


Fig. 4.11. Simulation Result for Initial Heading of $0°$, with Map Data 1

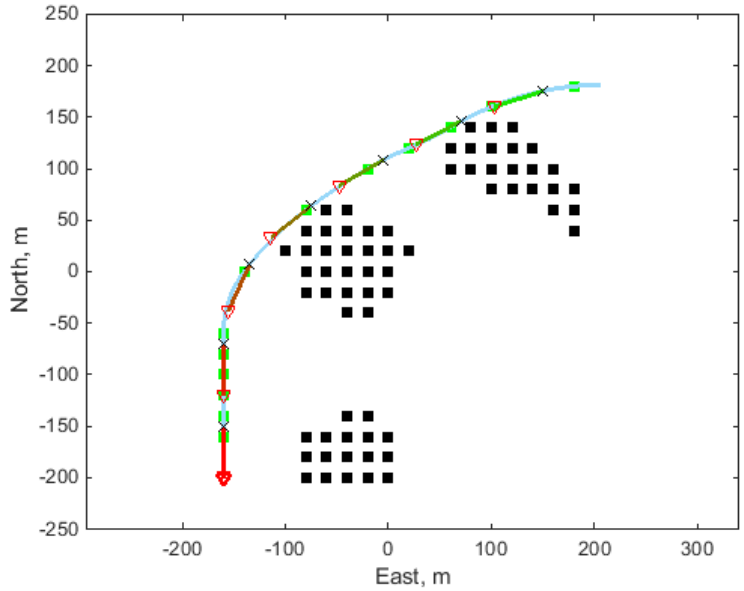


Fig. 4.12. Simulation Result for Initial Heading of $90°$, with Map Data 1

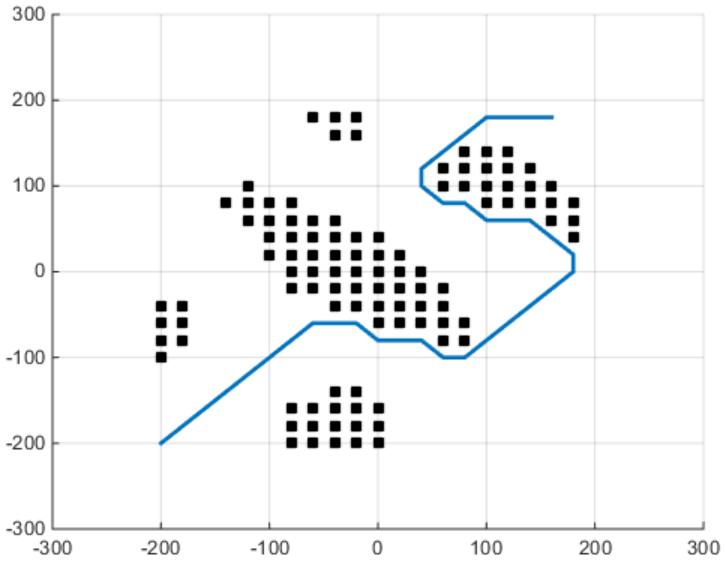


Fig. 4.13. Standard A* Search Result for Map Data 2

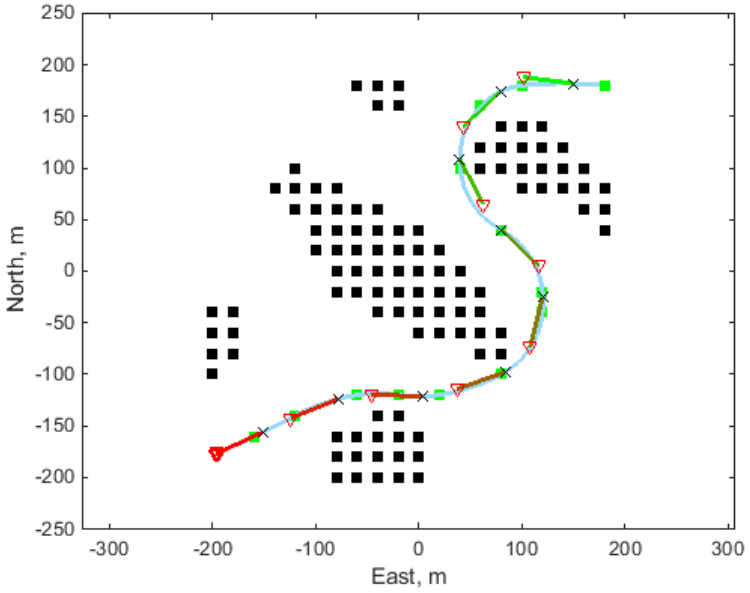


Fig. 4.14. Simulation Result for Initial Heading of $30°$, with Map Data 2

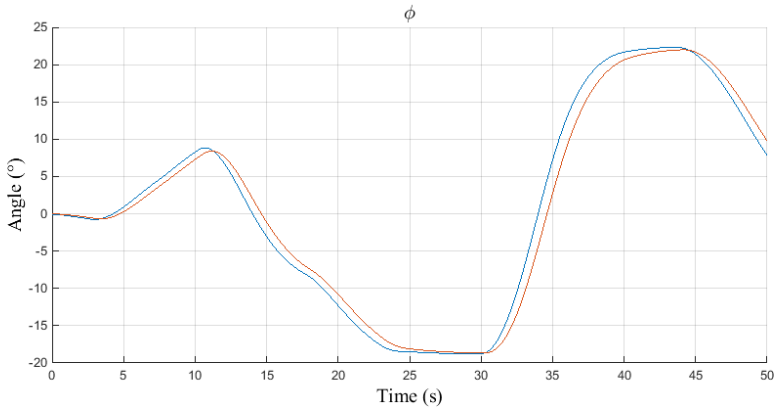


Fig. 4.15. ϕ Command and Reference of the Simulation of Map Data 2

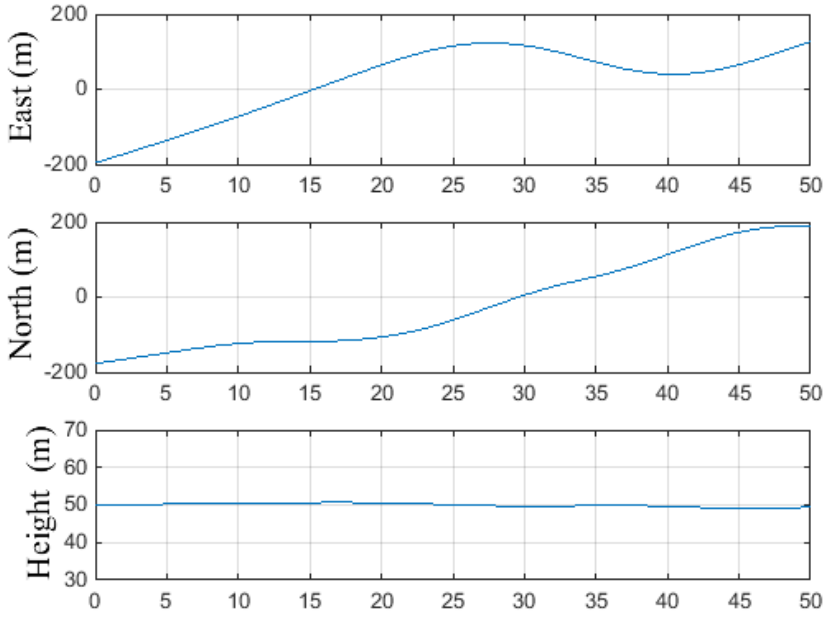


Fig. 4.16. Position of the simulation of Map Data 2

5. Flight Test

In this chapter, flight test result is shown. UAV system for the flight test will be briefly explained, and flight test preparation process and test result will be shown. In the flight test, same conditions used in Chapter 4 are used.

5.1 UAV System

5.1.1 UAV System Introduction

Skyscout model from Hitec Ltd. is used for the flight test. This model has EPO body that is endurance to impact. The wing span is 1.7m which is small enough for intermediate pilots as shown in Fig. 5.1. Main wing and tail wing can be detached for easy carrying, and electrical servo lines are mounted to body for convenient and robust connection. Fig. 5.2 shows the scene of the flight test. Several manual flights and stabilizing flights were pre-performed to understand the flight property of the UAV. Basic flight stability, responses about control commands, maximum climbing angles, and other various flight features were tested. In summary, the UAV model considered in this study is suitable for performing various flight test missions.

Fig. 5.1. Skyscout RC Aircraft

Fig. 5.2. Quality Test Experiment for Aircraft Model

UAV consists of (i) control section, which has throttle motor, servo motors, and control surfaces, (ii) sensor section, which has air pressure sensor, air velocity sensor, attitude, and GPS sensors, (iii) communication section, which has antennas and communication modules from and to the ground control system, (iv) flight computer system, which deals with computation and supervision, and (v) the body that attaches all these sections.

For the consistent dealing with the system, aircraft head for mounting the avionics was 3d scanned and modeled, and the head mount was manufactured with 3d printer to have same physical property as shown in Fig. 5.3 and Fig. 5.4. The head mount has space for various sensors and holes for antenna, especially, angle of attack sensor and sideslip angle sensor. Therefore robust guidance and control performance can be obtained even for the windy condition using these air data sensors.

Fig. 5.3. Head Mount by 3d Scanning

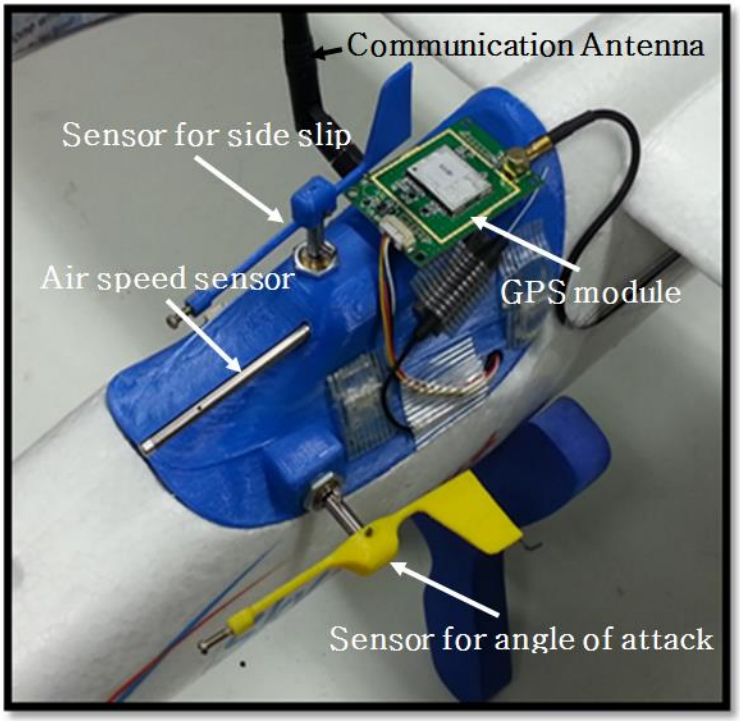


Fig. 5.4. Integrated Head Mount

5.1.2 **Flight Control Computer**

Flight control computer (FCC) is designed for various UAV missions. Small and integrated ARM based flight control computer is developed in this study. Flight control computer can deal with integrated sensors.

Table 5.1. Hardware Specification

Main processor	STM32F103VCT6
Telemetry	XBP-09-DMWIT-156
Airspeed sensor	MPXV7002
Barometric sensor	MS5611
IMU sensor	MPU9050
RC signal input	6 Channel
PWM signal output	6 Channel
ADC sensing	3 Channel
Status display	4 LED

Table 5.1 summarizes the specification of selected sensors and parts. The selected main processor a Cortex M3 series STM32F103 chipset. The chip is capable of computing guidance and control algorithm at 50Hz. RF module for communication is XBP-09-DMWIT-156 of Digi corp. This RF module has rapid response time, and it can communicate with other system in about 1km distance. Weight of the module is about 20g, with maximum data transmitting speed of 230,400bps. Air speed sensor is MPVX7002 differential pressure sensor. This sensor has resolution of $\pm 2kPa$. MPXV7002 sensor has two inputs, one is total pressure and the other is static pressure. Difference of these

two pressures is the dynamic pressure that comes out from the sensor. This sensor uses 5V level, so that the result is analog output of range 0~5V. It can sense about maximum 200km/h wind speed. Air speed is very important because GPS sensor cannot provide the information in the windy condition. Air pressure sensor is MS5611. It has a resolution to sense 10cm difference of air pressure. With this sensor, flight control computer can obtain the information of relative height in addition to GPS absolute height. This information is very important in the longitudinal control, because GPS sensor is not enough.

Flight control computer has MEMS inertial sensor to obtain its attitude information. The chip has 3-axis accelerometer, 3-axis angular velocity, and 3-axis magnetometer. Fig. 5.5 shows circuit design and integrated FCC.

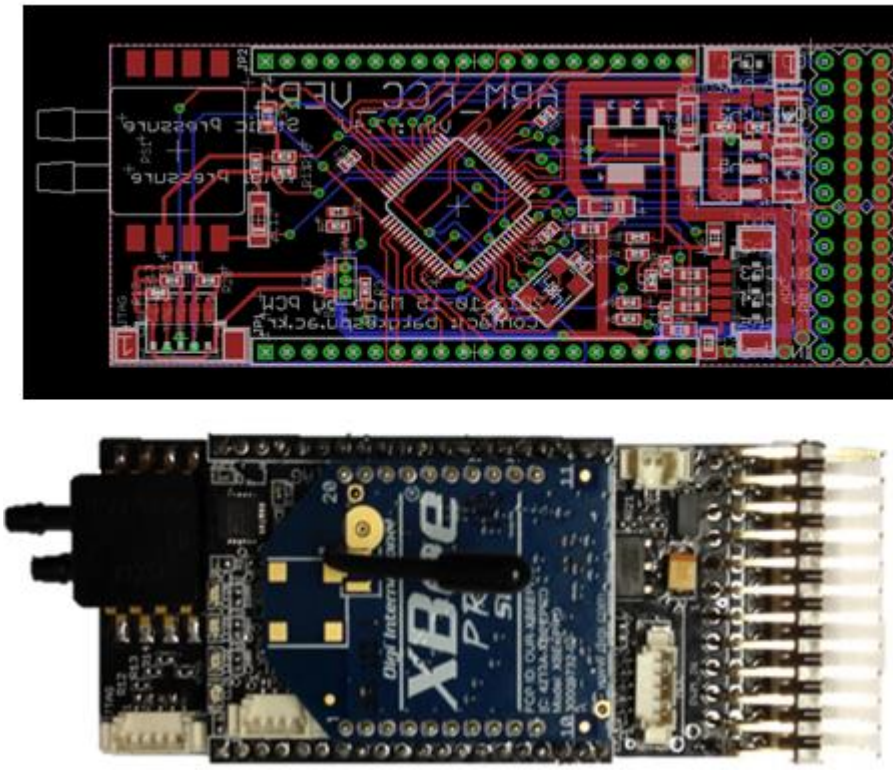


Fig. 5.5. Artwork and Flight Control Computer

5.2 Flight Test Preparation

Flight test site is selected that it has large vacant area for safety when UAV fails. In this study, reclaimed land near the fossil of dinosaur egg at Hwa-sung City, Kyungi-do is chosen. This site has been used for RC airfield for many years. It is covered with reed, and runway is exposed near airfields. There are no houses or villages near the site, but a highway is constructed recently. The test site has been moved to the inside of the region because of the highway as shown in Fig. 5.6.

Test site is composed of runway that UAV can land or take-off, an apron that UAV can wait sensors to ready before take-off, ground control tower for controlling and monitoring the UAV, recharge center for recharging the battery, and repair center for repairing small failure of the UAV.

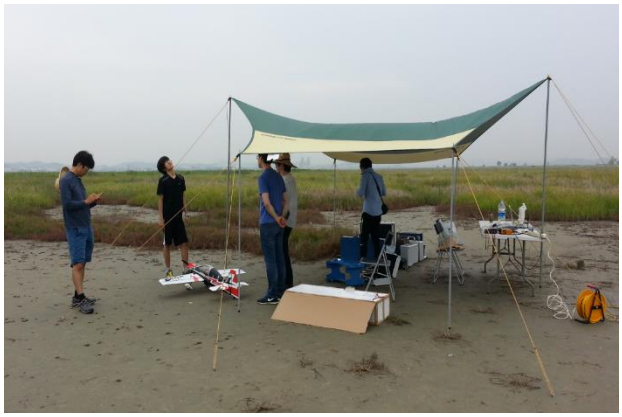


Fig. 5.6. Flight Test Site

5.3 Flight Test Result

Flight tests are performed using same initial values used in the numerical simulations. As in the numerical simulations, flight test scenario considers 2 maps and 3 different initial heading angles for comparing flight test result to simulation result.

Fig. 5.7 and Fig. 5.8 show flight test result of initial heading angle of 90 degrees. In the figures, triangle indicates UAV, and “x” symbol indicates the position of the calculated target position. Even though there exists incidence angle error, UAV flight path is converged immediately to the calculated flight path. State variables during the flight test are shown in Fig. 5.9 and Fig. 5.10.

Flight test result for initial heading angle of 0 degree is shown in Fig. 5.11 and Fig. 5.12. Again, UAV generates flight path for keeping away from the obstacles. State variables are shown in Fig. 5.13 and Fig. 5.14.

For the flight test result of initial heading angle of 30, same result is obtained. Flight test result and simulation result are shown in Fig. 5.15 and Fig. 5.16. Flight paths are generated considering bank angle limit, and the target position on the flight path is calculated. As shown in Fig. 5.15 and Fig. 5.16 the UAV follow the computed path well.

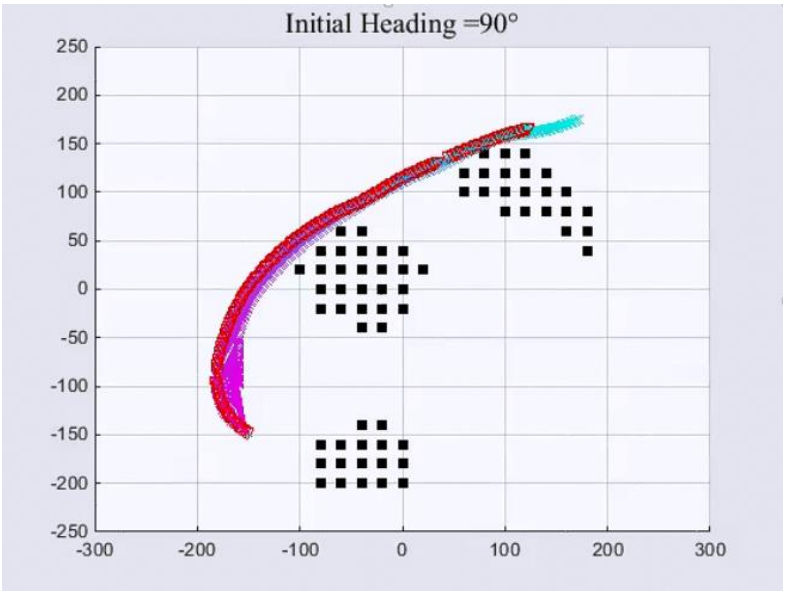


Fig. 5.7. Flight Test Result 1 of Scenario 1

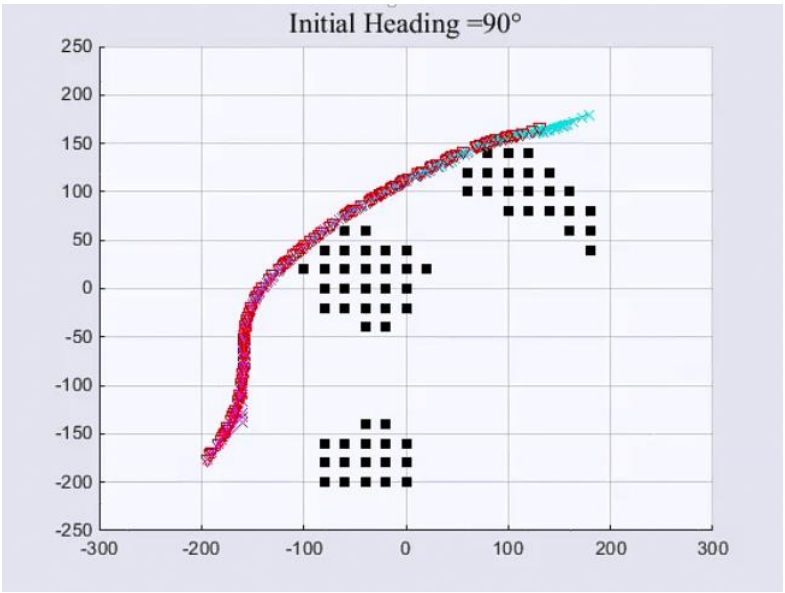


Fig. 5.8. Flgiht Test Result 2 of Scenario 1

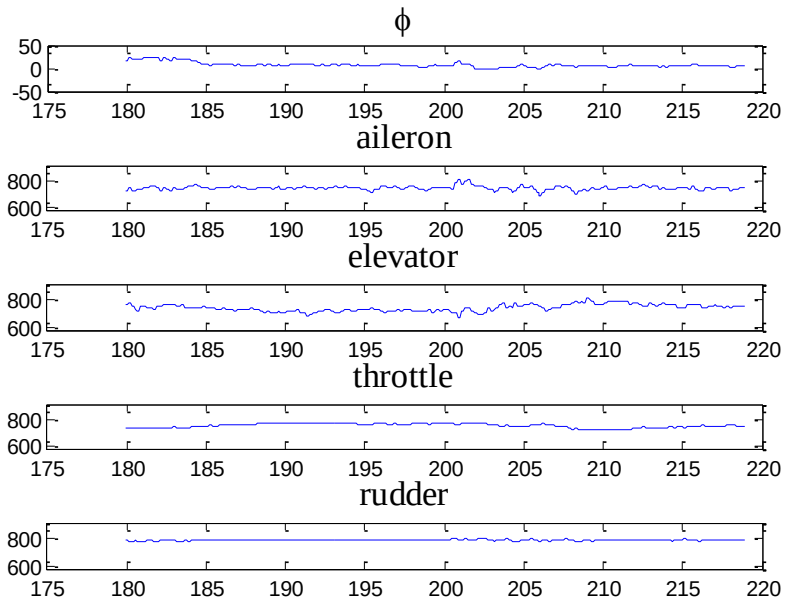


Fig. 5.9. Roll Angle and Actuator for Test Result 1 of Scenario 1

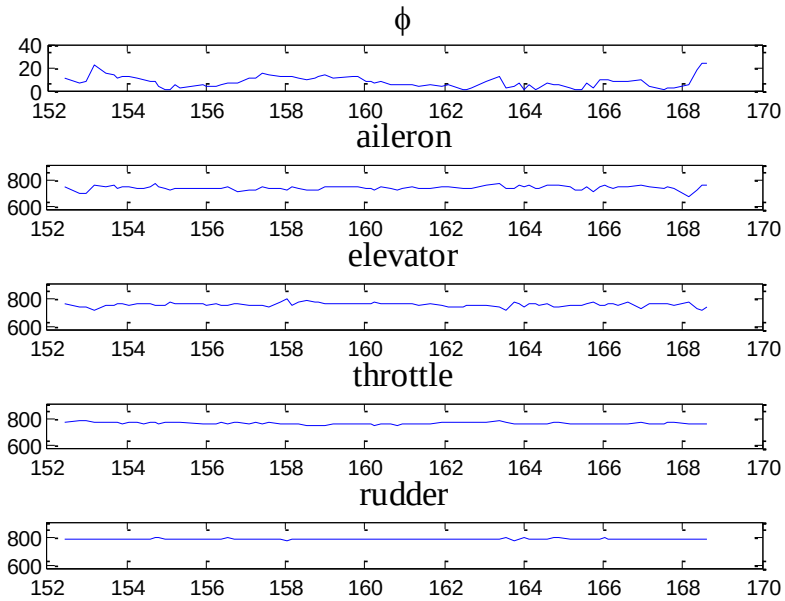


Fig. 5.10. Roll Angle and Actuator for Test Result 2 of Scenario 1

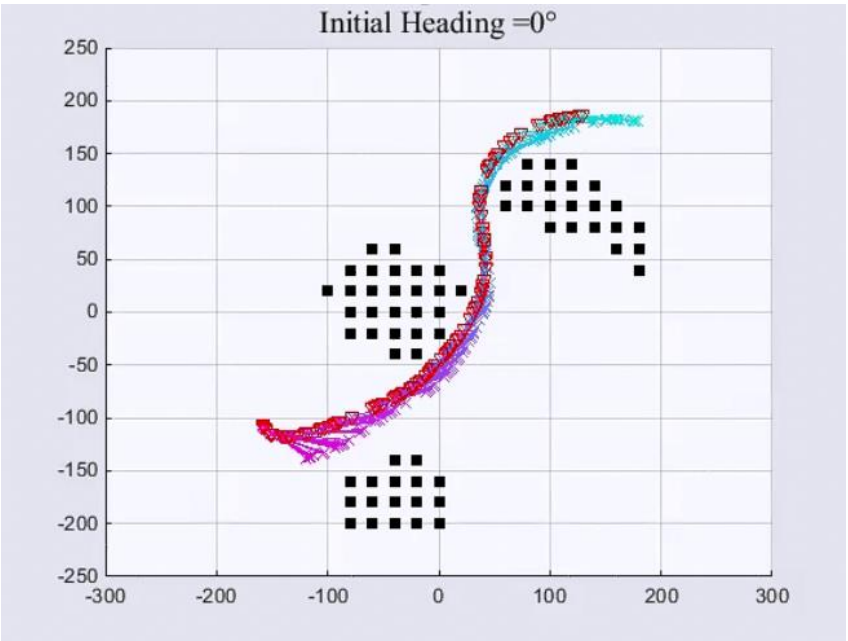


Fig. 5.11. Flight Test Result 1 of Scenario 2

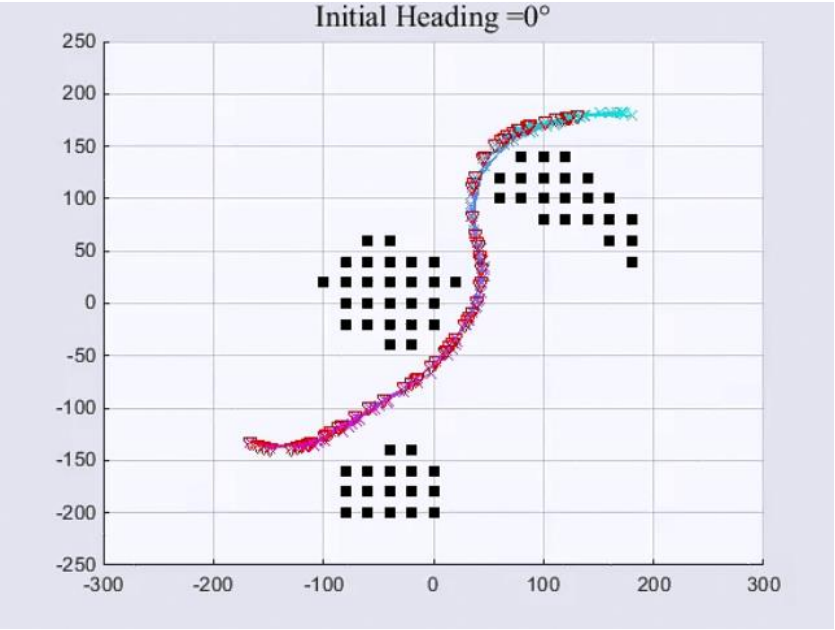


Fig. 5.12. Flight Test Result 2 of Scenario 2

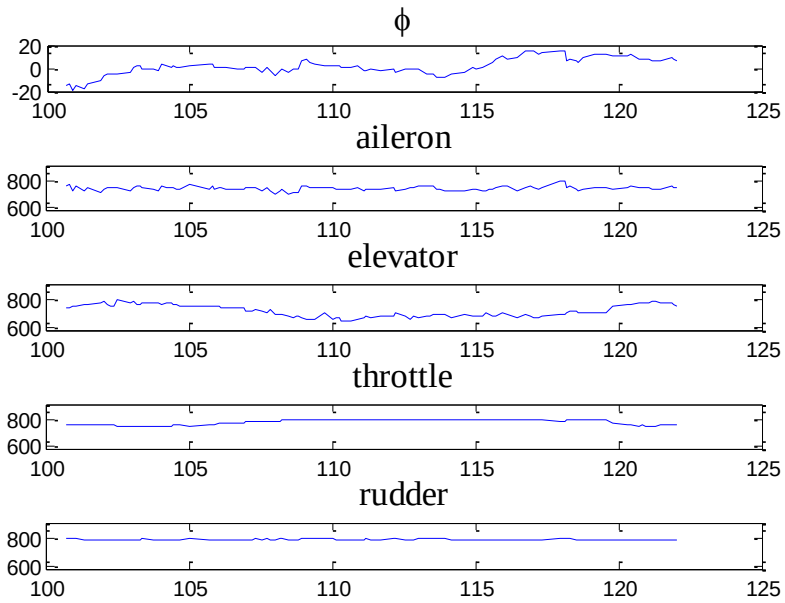


Fig. 5.13. Roll Angle and Actuator for Test Result 1 of Scenario 2

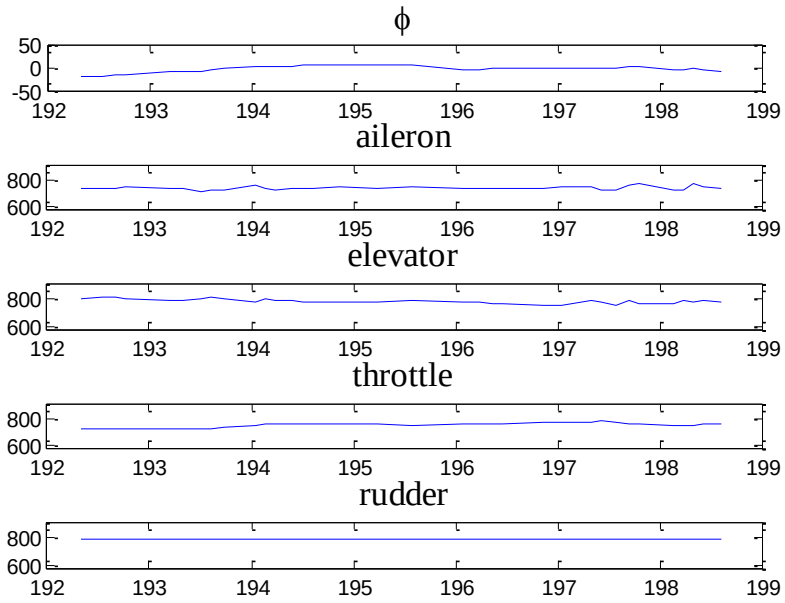


Fig. 5.14. Roll Angle and Actuator for Test Result 2 of Scenario 2

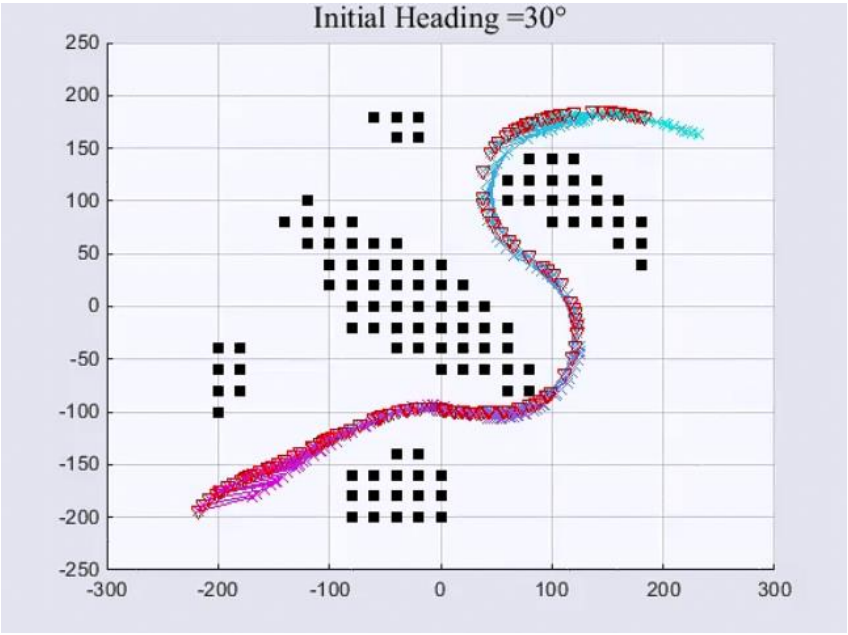


Fig. 5.15. Flight Test Result 1 of Scenario 3

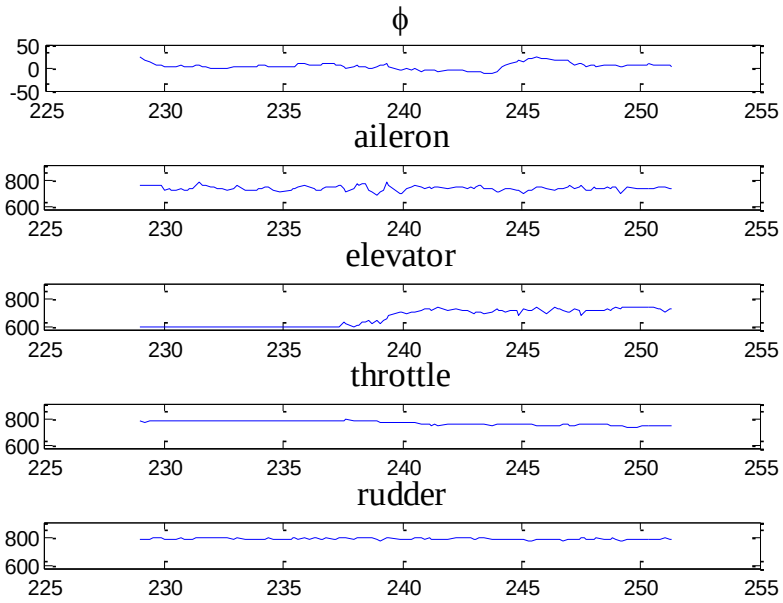


Fig. 5.16. Roll Angle and Actuator for Test Result 1 of Scenario 3

6. Conclusion

In this thesis, path planning algorithm for UAV to keep away from the prohibited area in real-time is proposed. A* path planning algorithm is modified to apply the algorithm to UAV considering the dynamics of the UAV. The effects of design parameters in the path designed by the path-planning algorithm are analyzed by simulation. The algorithm is applied to 6-axis linear simulation model. Also the proposed method is verified by flight tests. The proposed algorithm generates flight path for the UAV, and it does not need post-processing or smoothing of the computed path. The proposed method can be implemented in real-time using embedded computer on UAV.

Proposed algorithm is not a path planning method including reactive collision avoidance technic, because it uses pre-defined map data. Also the proposed method cannot be used for unknown regions. Therefore path planning algorithm considering (i) dynamic environment such as pop-up threat and (ii) uncertain region should be performed as future research.

Reference

[1] Zhu, H. G., Xin, H., and Zheng, C. W. "Research On UAV Path Planning," *Applied Mechanics and Materials* Vol. 58-60, 2011, pp. 2351-2355.

[2] Bortoff, S. A. "Path planning for UAVs," *American Control Conference, 2000. Proceedings of the 2000*. 6 ed. Vol. 1, 2000, pp. 364-368 vol.1.

[3] Pettersson, P. O., and Doherty, P. "Probabilistic roadmap based path planning for an autonomous unmanned helicopter," *Journal of Intelligent and Fuzzy Systems* Vol. 17, No. 4, 2006, pp. 395-405.

[4] Kavraki, L. E., Svestka, P., Latombe, J. C., and Overmars, M. H. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on* Vol. 12, No. 4, 1996, pp. 566-580.

[5] Khatib, O. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research* Vol. 5, No. 1, 1986, pp. 90-98.

[6] Dijkstra, E. W. "A note on two problems in connexion with graphs," *Numerische Mathematik* Vol. 1, No. 1, 1959, pp. 269-271.

[7] Hart, P. E., Nilsson, N. J., and Raphael, B. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *Systems Science and Cybernetics, IEEE Transactions on* Vol. 4, No. 2, 1968, pp. 100-107.

[8] Stentz, A. "Optimal and efficient path planning for partially-known environments," *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. 1994, pp. 3310-3317 vol.4.

[9] Ruz, J. J., Arevalo, O., Pajares, G., and de la Cruz, J. M. "Decision making among alternative routes for UAVs in dynamic environments," *Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on*. 2007, pp. 997-1004.

[10] Sanghyuk, P., John, D., and Jonathan, H. "A New Nonlinear Guidance Logic for Trajectory Tracking," *AIAA Guidance, Navigation, and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2004.

국문초록

무인항공기는 정찰, 감시 등의 군용임무를 위해 개발되기 시작하였으나, 최근에는 농업, 항공촬영 등의 민간분야에 활용되며 널리 확장되고 있다. 무인항공기의 운용에 있어서 위험지역 또는 장애물을 인지하고 회피하는 것은 반드시 선행되어야 하는 과제이다. 무인항공기가 위험지역으로 진입하거나 장애물과 충돌함은 시스의 파괴를 가져올 수 있으며, 나아가 인명피해 등의 손실을 입힐 수 있기 때문에 충돌회피 또는 위험지역 회피는 다양한 방법으로 연구가 수행되고 있다.

본 연구에서는 이와 같은 위협회피를 위하여 지상 로봇 연구분야에서 개발된 A* 경로계획 알고리즘을 고려하였다. 기존의 A* 경로계획 알고리즘이 무인항공기의 동역학적 특성을 반영하지 못하여 경로를 생성한 이후 후처리를 통해 사용할 수 밖에 없었다. 따라서 본 연구에서는 무인항공기 시스템에서 실시간으로 동작하면서 후처리 없이 회피기동을 수행할 수 있는 경로생성 기법을 제안하였다. 또한, 비선형 유도기와 PID 제어기를 이용하여 생성된 경로를 추종하는 유도제어기를 설계하였다. 비행시험을 통해 획득한 6 자유도 시뮬레이션 모델을 기반으로 성능을 검증한 뒤, 실제 비행시험을 통해 제안된 알고리즘의 성능을 검증하였다.

Keywords: 무인항공기, 충돌회피, A* 탐색법, 비선형 유도, 실시간 경로계획

Student number: 2013-20690