Small UAV Trajcetory Prediction and Avoidance using Monocular Computer Vision

Changkoo Kang

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Aerospace Engineering

Craig A. Woolsey, Chair
Seongim Choi, Co-Chair
Kyriakos G. Vamvoudakis

April 28, 2017
Blacksburg, Virginia

Keywords: Aircraft dynamics, Computer vision, Autonomous systems

Small UAV Trajcetory Prediction and Avoidance using Monocular Computer Vision

Changkoo Kang

(ABSTRACT)

Small unmanned aircraft systems (UAS) must be able to detect and avoid conflicting traffic, an especially challenging task when the threat is another small UAS. Collision avoidance requires trajectory prediction and the performance of a collision avoidance system can be improved by extending the prediction horizon. In this thesis, an algorithm for predicting the trajectory of a small, fixed-wing UAS using an estimate of its orientation and for maneuvering around the threat, if necessary, is developed. A computer vision algorithm locates specific feature points of a threat aircraft in an image and the POSIT algorithm uses these feature points to estimate the pose (position and attitude) of the threat. A sequence of pose estimates is then used to predict the trajectory of the threat aircraft and to avoid colliding with it. To assess the algorithm's performance, the predictions are compared with predictions based solely on position estimates for a variety of encounter scenarios. Simulation and experimental results indicate that trajectory prediction using orientation estimates provides quicker response to a change in the threat aircraft trajectory and results in better prediction and avoidance performance.

Small UAV Trajcetory Prediction and Avoidance using Monocular Computer Vision

Changkoo Kang

(GENERAL AUDIENCE ABSTRACT)

Small unmanned aircraft systems (UAS) must be able to detect and avoid conflicting traffic, an especially challenging task when the threat is another small UAS. Collision avoidance requires trajectory prediction and the performance of a collision avoidance system can be improved by extending the prediction horizon. In this thesis, an algorithm for predicting the trajectory of a small, fixed-wing UAS using an estimate of its orientation and for maneuvering around the threat, if necessary, is developed. A computer vision algorithm locates specific feature points of a threat aircraft in an image and a pose (position and attitude) estimation algorithm uses these feature points to estimate the pose of the threat. A sequence of pose estimates is then used to predict the trajectory of the threat aircraft and to avoid colliding with it. To assess the algorithm's performance, the predictions are compared with predictions based solely on position estimates for a variety of encounter scenarios. Simulation and experimental results indicate that trajectory prediction using orientation estimates provides quicker response to a change in the threat aircraft trajectory and results in better prediction and avoidance performance.

Acknowledgments

First and foremost, I would like to thank my Lord and Savior Jesus Christ who always gives me strength and wisdom. Second, I would like to thank my parents. My parents always encourage and support me to study at Virginia Tech and their help was one of the most important things throughout my master years. Third, I want to thank my advisor and mentor, Dr. Craig Woolsey. He has always given me thoughtful and kind advices so that I could learn numerous things from him to be a graduate student. Thanks to his perfect guidance, I could have developed my research well. Fourth, I also would like to express my gratitude to my co-advisor Dr. Seongim Choi and Dr. Kyriakos G. Vamvoudakis. Their kindness, suggestions and insight improved my understanding and broadened my vision. Lastly, I want to thank Hunter McClelland and Jason Davis. Hunter McClelland, who is a doctoral candidate in aerospace engineering, helped me with the flight test for the research and he gave me many advices. Also, Jason Davis, who is an undergraduate research assistant, helped me with developing the computer vision algorithm and his amazing work on computer vision has always surprised me. Without their help, I could not have completed this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, the private and commercial use of small UAS continues to expand and low altitude air traffic will become more congested raising the risk of mid-air collisions that may result in injuries to people or damage to property below. For many years, commercial manned aircraft have used the Traffic Collision Avoidance System (TCAS) to help ensure that aircraft do not collide in flight [1]. The TCAS and similar collision avoidance systems are useful, however, only when every aircraft in the airspace uses the technology. Small UAS typically operate at low altitude where collision threats include general aviation aircraft and other UAS, which may not include collision avoidance equipment [2]. In these scenarios, it is urgent that the small unmanned aircraft be able to sense and avoid these threats.

1.1 Sensor technologies

Sense and Avoid (SAA) technologies have been developed to mitigate the risk of collision between UAS and other aircraft and there are two main types of sensors that are used [3, 4]:

1. Cooperative types: Transponder [1, 5], Automatic Dependent Surveillance-Broadcast (ADS-B) [6]

2. Non-cooperative types: Radar [7], LiDAR [8], Optical sensor [9, 10, 11]

Cooperative types depend on communication between aircraft. Therefore, required equipment for communication must be installed on each aircraft. The TCAS, for example, requires a transponder. The TCAS is most prevalent SAA sensor since all current commercial aircraft are required to equip this system in controlled airspaces. The TCAS exchanges the information using the Traffic Advisory (TA) and the Resolution Advisory (RA) using transponders. The TA is a spoken message, "traffic, traffic." This message is generated if a collision is predicted within 20 to 48 seconds. The RA gives an aural command such as "climb, climb" and a graphical display of the intruder vertical rate [1]. The TCAS also has a long range of view up to 160km. However, the weakness of the TCAS is that it cannot communicate with other aircraft which do not have the required equipment for communication. Also, transponder is inadequate for small UAS because of the size and required power. Another cooperative method, the ADS-B is a broadcast technology. The ADS-B broadcasts its position, altitude and velocity automatically with a long range (up to 240km). In contrast with transponders, small and light ADS-B systems have been developed, hence the ADS-B can be carried by more aircraft than the TCAS. However, the other aircraft also need to carry this equipment to receive the signal and the ADS-B technologies are not widely used yet.

Non-cooperative types do not require active communication with other aircraft. Therefore, aircraft do not require common equipment for communication. However, non-cooperative methods are less accurate relatively than cooperative methods. Radar is a prevalent non-cooperative sensor. This sensor actively senses obstacles and provides the location, velocity and size of these obstacles. Also, recent development of radar technology makes the size, weight and power and cost (SWaP-C) of radar low enough that radar may be integrated into small UAS [12]. LiDAR gives the range to obstacles by analyzing reflected light. However, the weakness of LiDAR is that it has a limited field of view (FOV) and it has shorter range

of view (3km) than other sensors. The camera sensor detects the obstacles using natural visible light. It provides an azimuth angle and elevation angle to the obstacle. The SWaP-C is also low since the technology has been advanced thanks to intense competition within the consumer electronics industry. Therefore, the camera sensor also can be used for a small UAS. However, camera sensors have relatively short range of view and detection performance decreases sharply in low light conditions. Also, visual clutter is generated by sunlight or clouds can degrade detection performance as well.

Using such sensors, SAA systems detect and avoid threat aircraft by estimating the threat's position and velocity and predicting its motion. The approach is especially effective if the threat aircraft flies on a straight path. However, the trajectory prediction based on this data is less accurate when the threat aircraft is turning. With the development of camera technologies and computer vision technologies, camera sensors are capable of detecting the orientation of the threat aircraft which provides additional information that can be used in predicting the threat's trajectory.

1.2 Computer vision technology

An image is a matrix of pixels and each element of the matrix is a specific number which indicates the color of the pixel. Using these numbers, an image can be modified and analyzed by computer vision technologies. In order to detect the orientation of the threat aircraft using a camera, computer vision technologies are necessary. First, the threat aircraft needs to be detected. For the detection and tracking, there are several computer vision technologies.

1. Edge detection [13, 14]
2. Template matching [15]

3. Optical flow [16, 17, 18]

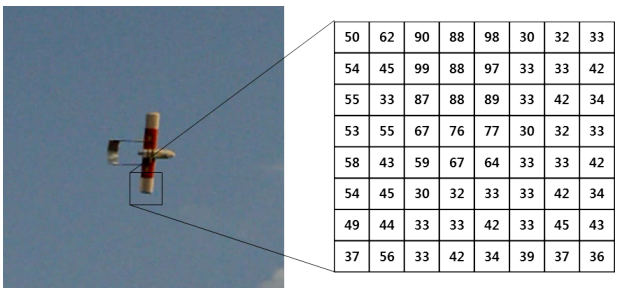


Figure 1.1: Image matrix

1.2.1 Edge detection

The edge detection algorithm uses numbers in an image matrix to find the edge. In an image matrix, an edge means a 'sudden change' of numbers in the matrix. As shown in Figure 1.1, the number of the third column on the first low of the matrix are suddenly increased to 90 from 62 of the second column. The edge detection algorithm considers these 'sudden changes' as an edge and the size of this 'sudden change' is determined as a threshold value by users. The edge detection algorithm finds edges in the image and change the detected pixels to 1 and other pixels become 0. These pixels whose color differs significantly from that of neighboring pixels are determined by a threshold value and the image becomes a binary image which shows edges of the image. However, using only gradient, there can be many false alarms, hence researchers have suggested various approaches to make edge detection performance more accurate.

Several edge detection techniques have been introduced and developed:

1. Classical gradient edge detection [14]

2. Zero crossing [13]

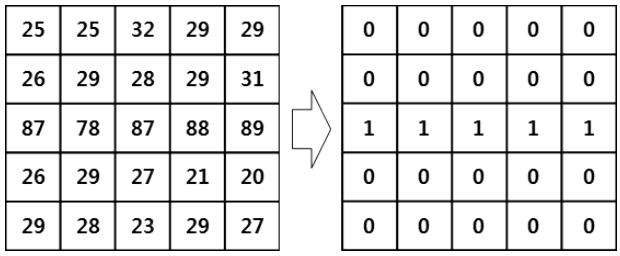


Figure 1.2: Edge detection process

3. Laplacian of Gaussian (LOG) [14]

4. Gaussian Edge Detectors [13]

The classical gradient edge detection technique uses the first derivative operation; thus, the algorithm is simple to implement, but sensitive to noise and not accurate. The Sobel operator and Prewitt operator are examples of this technique. The zero crossing uses the second derivative operation so that it can also find the orientation of edges, but this technique is also sensitive to the noise. The LOG uses the Laplacian after applying the Gaussian Smoothing on an image. Therefore, it is less sensitive than the aforementioned edge detection techniques. However, it cannot find the orientation of the edges and has some errors at corners and curves. Lastly, the Gaussian edge detectors use the gradient edge detection after the Gaussian Smoothing of an image. The Canny edge detector is known as the most popular Gaussian edge detector. This technique works better than other techniques in noise conditions [14].

1.2.2 Template matching

The template matching algorithm is a technique to find a subset of pixels whose properties (color, shape, etc.) match a pre-defined pattern, or template, in an image. The template matching algorithm slides the template over the search area of the source image and measure

the similarity between the source image and template at each position of the source image. Similarity is an indicator how much the source image I and the template T are similar and the similarity is determined with a similarity value. There are various approaches to estimate the similarity:

1. Sum of Squared Differences (SSD) and Normalized Sum of Squared Differences (NSD) [19, 20]

2. Cross-Correlation (CC) and Normalized Cross-Correlation (NCC) [21]

3. Correlation Coefficient (CO) and Normalized Correlation Coefficient (NCO) [22]

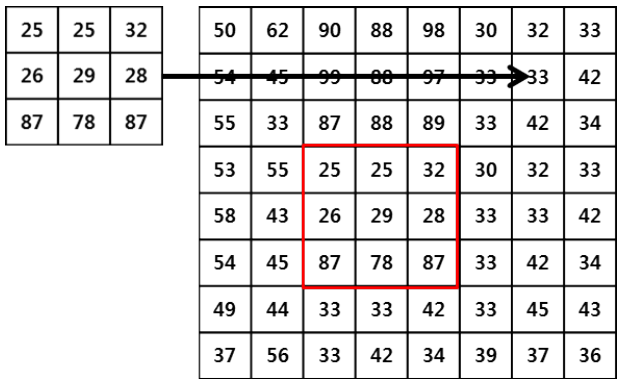


Figure 1.3: Template matching process

The SSD and the NSD values compute squared differences between I and T. The value of the source image I at position (x,y) on the source image is described as $I(x, y)$ and the value of the template image T at position (i,j) on the template image is described as $T(i, j)$ with the template width w and height h using the equation:

$$
\begin{aligned}
SSD(x,y) &= \sum_{j=0}^{N}\sum_{i=0}^{M}(T(i,j) - I(x+i,y+j))^2 \\
NSD(x,y) &= \frac{\sum_{j=0}^{N}\sum_{i=0}^{M}(T(i,j) - I(x+i,y+j))^2}{\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}T(i,j)^2}\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}I(x+i,y+j)^2}}
\end{aligned}
\tag{1.1}
$$

where M is the number of rows of the template image and N is the number of column of the template image. The CC value and the NCC value can be computed by:

$$\begin{aligned} CC(x,y) &= \sum_{j=0}^{N}\sum_{i=0}^{M}(T(i,j)I(x+i,y+j)) \\ NCC(x,y) &= \frac{\sum_{j=0}^{N}\sum_{i=0}^{M}(T(i,j)I(x+i,y+j))}{\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}T(i,j)^2}\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}I(x+i,y+j)^2}} \end{aligned} \tag{1.2}$$

Lastly, the CO value and the NCO value are obtained by:

$$\begin{aligned} CO(x,y) &= \sum_{j=0}^{N}\sum_{i=0}^{M}(T'(i,j)I'(x+i,y+j)) \\ NCO(x,y) &= \frac{\sum_{j=0}^{N}\sum_{i=0}^{M}(T'(i,j)I'(x+i,y+j))}{\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}T'(i,j)^2}\sqrt{\sum_{j=0}^{N}\sum_{i=0}^{M}I'(x+i,y+j)^2}} \end{aligned} \tag{1.3}$$

where

$$T'(i,j) = T(i,j) - \frac{1}{wh}\sum_{j=0}^{N}\sum_{i=0}^{M}T'(i,j)$$

$$I'(x+i,y+j) = I'(x+i,y+j) - \frac{1}{wh}\sum_{j=0}^{N}\sum_{i=0}^{M}I'(x+i,y+j)$$

In [15], the accuracy and the processing time of these six algorithms are compared. The accuracy and the processing time of these methods are not significantly different; however, normalized based methods and the SSD shows more accurate results than the CC and CO. Also, the SSD has the fastest processing time.

1.2.3 Optical flow

The purpose of the optical flow algorithm is to estimate the transitional position change of pixels in an image. For this process, the optical flow algorithm tracks pixels between two images and computes the translation vectors of the pixels.

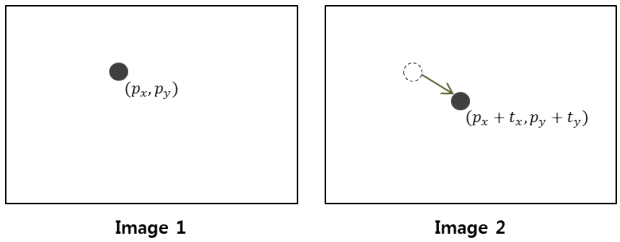


Figure 1.4: Optical flow process

For example, a source vector from the first image is defined as:

$$v_{src} = (p_x, p_y)^T \tag{1.4}$$

A destination vector which is located by the algorithm in the second image is:

$$v_{dst} = (p_x + t_x, p_y + t_y)^T \tag{1.5}$$

where p_x and p_y are the original pixel coordinates in the first image and t_x and t_y describe the translation of the pixel from its prior coordinate location in the second image. Therefore, the pixel velocity is described with the translation vector and this vector is visualized as a vector by the optical flow algorithm. This technique can be used to find a moving object in an image since a moving object has a pixel velocity in the image. Optical flow can be implemented by several methods and each method has different approaches to find the same pixels in the two different images:

1. Sum of Squared Differences (SSD) and Normalized Cross-Correlation (NCC)
2. Gunnar Farneback Algorithm [23]
3. Lucas-Kanade algorithm [24]

As mentioned, the first step of the optical flow is tracking pixels. Therefore, the template matching algorithm can be used for this process. The SSD and NCC which are used for the template matching algorithm can be used to track the pixels between the two images. However, the computation is processed for the entire image, hence this algorithm is computationally expensive and not efficient. Gunnar Farneback Algorithm uses quadratic polynomials to compute the movement between two images using the polynomial expansion transform. However, it can be also computationally expensive since it computes for all pixels in an image. Lucas-Kanade algorithm takes advantage of the fact that the two consecutive images are close to each other. Given functions $F(X)$ and $G(X)$ which indicate the values of pixels at the location vector $X = [x, y]$, this algorithm firstly finds a disparity vector h that minimize difference between $F(X + h)$ and $G(X)$ in a region of interest R. By doing this, the computation cost can be cheaper than other methods. The disparity vector h can be computed by:

$$\begin{aligned} &h_0 = 0, \\ &h_{k+1} = h_k + \frac{\Sigma_X w(X) F(X + h_k)[G(X) - F(X + h_k)]}{\Sigma_X w(X) F(X + h_k)^2} \\ &w(X) = \frac{1}{|G(X) - F(X)|} \end{aligned} \tag{1.6}$$

1.3 Aircraft collision avoidance

1.3.1 State estimate propagation

For aircraft collision avoidance, the state estimate propagation for the threat aircraft is important since the avoidance control depends on the predicted position of the threat aircraft. There are four approaches of aircraft state propagation [25, 26]:

1. Straight projection
2. Worst case projection
3. Probabilistic projection
4. Flight plan sharing

The straight projection propagates the state estimate in a single straight line and it does not consider about the uncertainty which all sensors have or possible direction changes of the threat aircraft. Therefore, the prediction computation is simple, but it can be used only for the situation that the future trajectory is predictable or for the short period of time.

While the straight projection treats a nominal trajectory, the worst case projection considers all possible trajectories of the aircraft. Therefore, this approach makes a range which consists of all possible future state estimate in a time horizon of the aircraft; and if there is a possibility for an obstacle to be in this range, a conflict is predicted. Therefore, the worst case projection can provide safer and more conservative avoidance results than the straight projection. However, this approach is more computationally expensive since it computes all future state estimate.

The probabilistic projection considers the uncertainties of future trajectories and creates a probability density function of the future states of the threat aircraft. Therefore, an

avoidance maneuver is determined based on the probability of collision. Also, the safety value and the false alarm rate can be analyzed based on the probability. The disadvantage of this approach is that it is computationally expensive and it is not easy to obtain a proper model for the probability of future state estimate of the aircraft.

The flight plan sharing works based on the shared information between aircraft. TCAS and ADS-B use this approach to avoid the threat. This method is the most efficient approach if it is possible since the host aircraft can receive all the information about the threat aircraft such as the position, heading angle, velocity and so on. However, as previously mentioned, this information sharing is possible when both aircraft have specific equipments to communicate each other.

1.3.2 Collision avoidance methods

After the trajectory prediction process, avoidance algorithms are also needed. In [27, 28], several avoidance methods are introduced:

1. Prescribed resolution methods
2. Optimization methods
3. Geometric methods
4. Potential field methods

The prescribed resolution methods return a predefined avoidance maneuver when the possible conflict is predicted. Therefore, the response time is very short since additional complex computations are not required. However, because of the simplicity, the effectiveness and optimality is decreased and this methods cannot react to an unexpected situation.

The optimization methods provide the optimal escape trajectories with the lowest cost.

With these methods, the optimal escape trajectories are computed with a kinematic model and a set of constraints. There are a variety of approaches to do this process such as a game theory approach, dynamic window approach, genetic algorithm, expert systems and fuzzy logic techniques. The optimization methods decrease the cost of the aircraft avoidance maneuver so that the avoidance maneuver can be efficient. However, the computation is complicated and time consuming.

The geometric methods compute the avoidance maneuver using geometric properties such as the position and velocity of the threat aircraft. These methods are less computationally expensive than the optimization methods; however there can be some deviation from the original trajectories while the optimization methods can minimize the deviation.

Lastly, potential field methods use a simple electrostatic equation to compute safe trajectories. These methods consider the threat aircraft as a charged particle and the repulsive forces from the force field equation are applied between two aircraft. These methods are adequate for local collision avoidance with a small number of obstacles whose position is exactly known. However, some sharp discontinued control can be generated and these methods treat only the state information of the threat aircraft and do not consider dynamic limitations, hence an error on the state estimate can cause a control error.

1.4 Thesis organization

This thesis is organized as follows. Chapter 2 describes the computer vision methods and various computer vision technologies are implemented. In Chapter 3, a pose estimation algorithm based on the feature points from the computer vision technology is introduced. Also, the pose estimation experiment and results are presented. Chapter 4 describes the aircraft motion model and two approaches to prediction: the proposed approach and a more

conventional approach for comparison. A simulation and an experiment are implemented to compare the performances of suggested trajectory prediction approaches. Chapter 5 develops an avoidance algorithm and this algorithm is also implemented in a simulation. Finally, Chapter 6 presents conclusions and summarizes the future work.

Chapter 2

Computer Vision Technology

2.1 Introduction

Recent digital cameras have a human face auto-focus function which is one of computer vision techniques. It detects human's face in the image and uses it as a focus point. Also, autonomous driving cars are using computer vision technologies to detect and avoid obstacles in front of them. Likewise, computer vision technologies have become necessary in our life and they will be more popular in the future as well.

Also, cameras have low size, weight, and power and cost (SWaP-C) as mentioned in Section 1.1, cameras can be useful sensors for small UAS which cannot afford to use heavy and power hungry sensors. For this research, a camera images a small UAS and computer vision technologies are used to analyze the orientation of a fixed-wing threat aircraft in camera images. These research efforts are performed in collaboration with Hunter McClelland, a doctoral candidate in Virginia Tech's Aerospace & Ocean Engineering program, who managed flight tests of this research and Jason Davis, an undergraduate research assistant in

Virginia Tech's Aerospace & Ocean Engineering program, who developed an edge detection algorithm and a pentagon optimization algorithm.

2.2 Computer Vision image processing

In order to compute the orientation of a threat aircraft in an image, a computer vision technology is used to detect five distinguished feature points of the threat aircraft: the nose, the two wing tips, and the two tips of the horizontal stabilizer. These points are assumed to be visible in the image and the geometry of the threat aircraft is known. Once the feature points are found, a pose estimation algorithm using these feature points is implemented to compute the rotation matrix R_{BI} that maps free vectors from the inertial reference frame to the aircraft body frame. From this matrix, Euler angles of the threat aircraft in the inertial frame can be computed. This pose estimation algorithm is described in detail in the next chapter. For all these processes, OpenCV library [29, 30] and a dataset from flight tests are used [12].

2.3 Flight experiments

The computer vision algorithm and the pose estimation algorithm are applied to experimental data obtained during flight tests. Experimental data were obtained during a Summer 2016 flight test campaign that was aimed at creating a database of visual and radar encounter data for several small UAS [12]. The flight tests were implemented at the Kentland Experimental Aerial Systems (KEAS) Laboratory in Blacksburg, VA, USA.

2.3.1 Hardware

For this research, the threat aircraft is custom-designed at Virginia Tech, named the electric Small Platform for Autonomous Aerial Research Operations (eSPAARO), shown in Figure 2.1. A PixHawk controller is mounted on the eSPAARO as a flight control system. The PixHawk contains an IMU, GPS, airspeed sensor, so that the position and velocity data of the threat aircraft are obtained from these sensors. Also, a Piksi Real Time Kinematic (RTK) GPS unit, which provides centimeter-level positioning accuracy, is added to the flight control system.

For a ground camera system, the Nikon D3200 digital camera used to image the eSPAARO was mounted on a tri-pod, together with a second PixHawk unit, located on the ground; future flight tests will involve air-to-air imagery.

Figure 2.1: eSPAARO and ground camera system

2.3.2 Experimental data

In the tests described above, camera images of the threat aircraft were obtained as shown in Figure 2.2. Also, a camera orientation log from the PixHawk of the ground camera system

and a position and velocity data of the eSPAARO from the PixHawk were also obtained.

Figure 2.2: Images from ground camera system

TimeUS	Roll	Pitch	Yaw	Alt	Lat
6.56E+08	1.75	15.27	126.31	532.79	37.19
6.56E+08	1.76	15.14	126.43	533.03	37.19
6.56E+08	1.77	15.04	126.54	533.03	37.19
6.56E+08	1.79	14.95	126.68	533.03	37.19
6.57E+08	1.79	14.85	126.9	532.78	37.19
6.57E+08	1.8	14.76	127.14	532.78	37.19
6.57E+08	1.82	14.69	127.36	532.7	37.19
6.57E+08	1.85	14.63	127.58	532.7	37.19
6.57E+08	1.88	14.57	127.77	532.7	37.19
6.57E+08	1.92	14.52	127.94	532.83	37.19
6.57E+08	1.96	14.46	128.1	532.83	37.19
6.57E+08	1.96	14.4	128.2	532.93	37.19
6.57E+08	1.94	14.28	128.29	532.93	37.19
6.57E+08	1.9	14.13	128.38	532.93	37.19
6.57E+08	1.89	13.89	128.46	532.71	37.19
6.57E+08	1.91	13.64	128.54	532.71	37.19
6.57E+08	1.93	13.44	128.62	532.74	37.19
6.57E+08	1.94	13.28	128.69	532.74	37.19
6.57E+08	1.97	13.15	128.78	532.74	37.19

Line #	TimeUS	Roll	Pitch	Yaw	Alt	Lat	Lng	GPS Time
84281	5.14E+08	36.32	-4.09	251.16	614.93	37.19889	-80.569	2016-08-06 11:06:18.021
84307	5.14E+08	35.48	-3.9	251.69	614.93	37.19889	-80.569	2016-08-06 11:06:18.062
84353	5.14E+08	34.66	-3.7	252.22	614.49	37.19889	-80.569	2016-08-06 11:06:18.101
84376	5.14E+08	33.77	-3.51	252.75	614.49	37.19889	-80.569	2016-08-06 11:06:18.141
84410	5.14E+08	32.82	-3.29	253.26	614.4	37.19889	-80.569	2016-08-06 11:06:18.182
84440	5.14E+08	31.79	-3.12	253.72	614.4	37.19888	-80.5691	2016-08-06 11:06:18.221
84466	5.14E+08	30.78	-2.92	254.17	614.4	37.19888	-80.5691	2016-08-06 11:06:18.262
84504	5.14E+08	29.63	-2.7	254.59	614.19	37.19888	-80.5691	2016-08-06 11:06:18.301
84527	5.15E+08	28.2	-2.48	254.97	614.19	37.19888	-80.5691	2016-08-06 11:06:18.340
84558	5.15E+08	26.66	-2.25	255.33	614.35	37.19888	-80.5691	2016-08-06 11:06:18.382
84589	5.15E+08	24.91	-2	255.68	614.35	37.19887	-80.5691	2016-08-06 11:06:18.424
84615	5.15E+08	23.16	-1.78	255.99	614.35	37.19887	-80.5691	2016-08-06 11:06:18.462
84653	5.15E+08	21.29	-1.64	256.29	613.07	37.19887	-80.5691	2016-08-06 11:06:18.501
84676	5.15E+08	19.31	-1.47	256.56	613.07	37.19887	-80.5691	2016-08-06 11:06:18.540
84708	5.15E+08	17.25	-1.28	256.84	613.66	37.19887	-80.5691	2016-08-06 11:06:18.582
84743	5.15E+08	15.06	-1.13	257.09	613.66	37.19886	-80.5692	2016-08-06 11:06:18.624
84773	5.15E+08	12.77	-0.96	257.35	613.66	37.19886	-80.5692	2016-08-06 11:06:18.662
84818	5.15E+08	10.56	-0.78	257.6	613.1	37.19886	-80.5692	2016-08-06 11:06:18.701

Figure 2.3: Camera orientation data (left) and eSPAARO log data (right)

2.4 Feature points detection

Using images from the ground camera system, various computer vision algorithm are applied to obtained images from flight tests. There are many ways to detect the threat aircraft: color detection, optical flow, template matching and edge detection. In this section, these computer vision algorithms are implemented and compared.

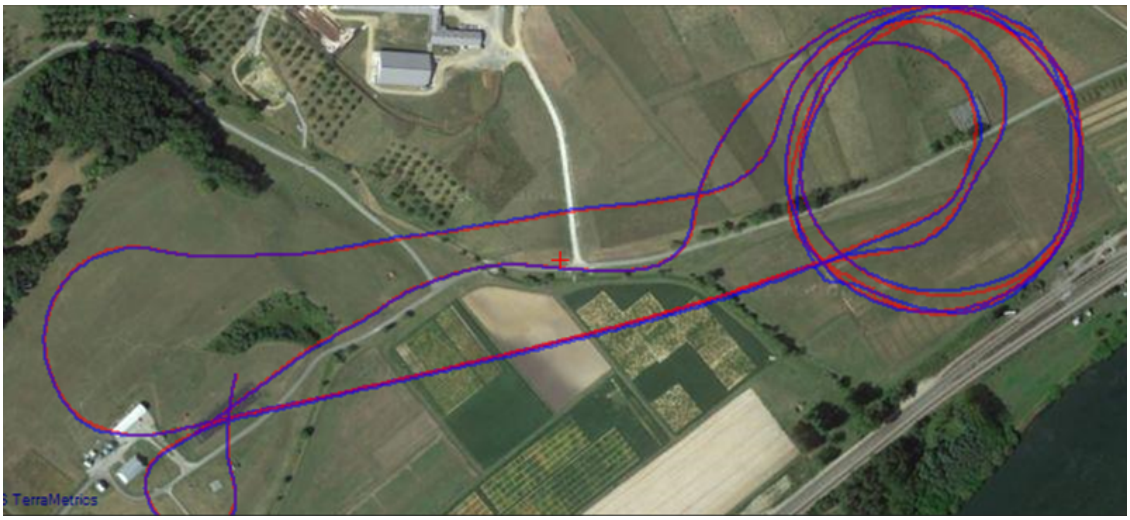


Figure 2.4: Entire aircraft trajectory of the flight test campaign

2.4.1 Color detection

The color detection algorithm analyzes the hue, saturation and value (HSV) of images. The image is firstly filtered by OpenCV function [29, 30] in the color detection code with the threshold HSV range values and the contour of the filtered image is obtained.

Figure 2.5: Color detection

With HSV ranges, the specific colors can be extracted as shown in Figure 2.5. In this image, the threat aircraft has a different color with the background. Therefore, the color detection algorithm could detect the aircraft accurately.

Figure 2.6: False alarm occurred by the background color

However, as shown in the Figure 2.6, the color detection algorithm detects even the background since the color of the background and the aircraft is similar. Therefore, the threat aircraft can be detected when the color of the threat aircraft is known and the background color is also different. This method is computationally less expensive and finds the target well when the image satisfies the described conditions. However, the color of the image can be easily affected by the light conditions; therefore, the detection is not consistent.

2.4.2 Optical flow

The optical flow algorithm tracks pixels between two images and computes the translation vectors of the pixels. Therefore, a moving object can be detected using this method since the position of the moving object within the next image will be different from the position within the previous image while the position of the static object in the image is almost the same in two consecutive images. For the implementation, the Lucas-Kanade algorithm is used.

The optical flow algorithm can find the moving object well in the image. Also, it can find the moving object consistently if the object is continuously moving. However, if the camera is moving, the false alarm rate of the optical flow algorithm becomes higher since even the

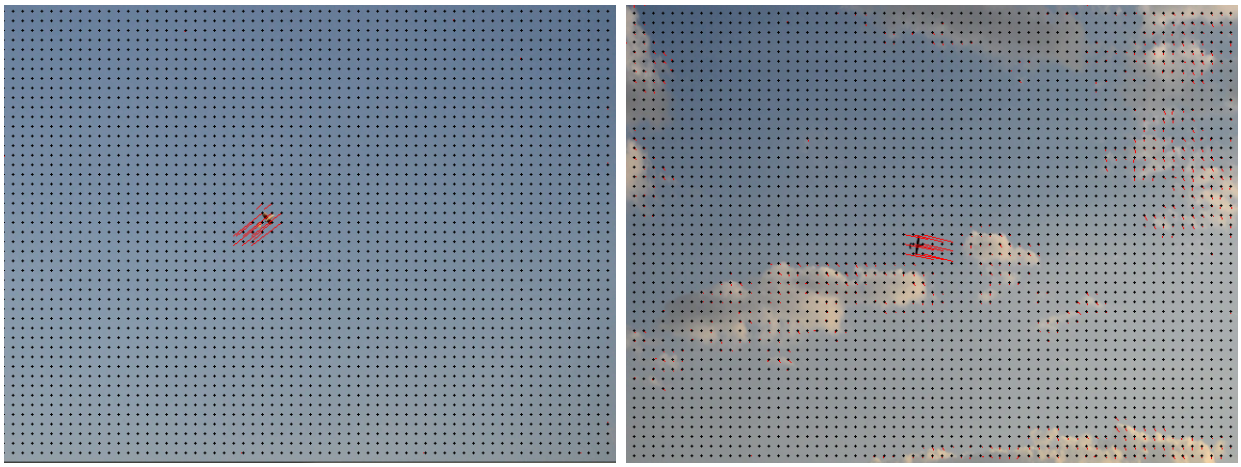


Figure 2.7: Optical flow

background is also moving so that recognizing the moving target is hard. Also, the optical flow algorithm is relatively computationally expensive since it computes translation vectors of each pixel.

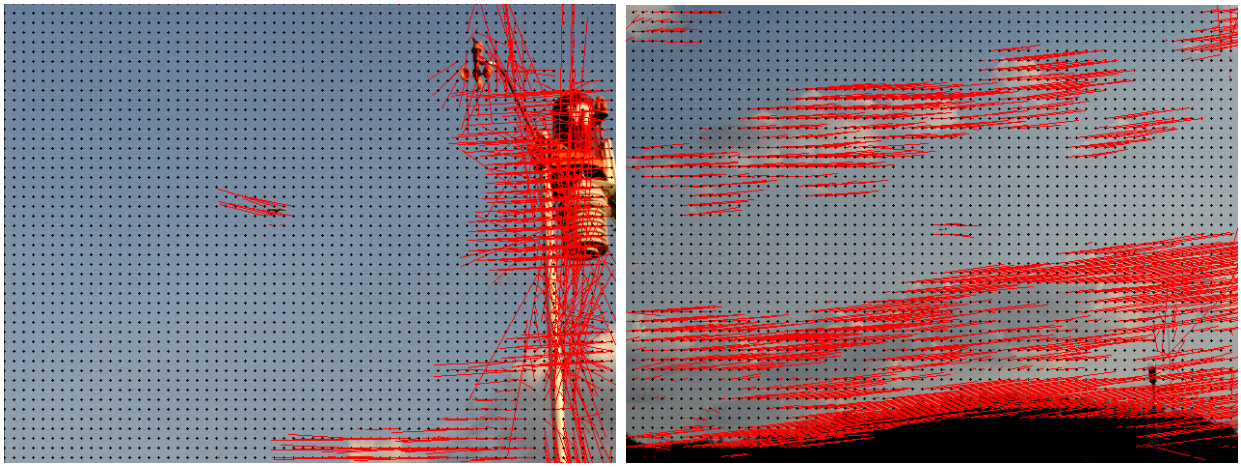


Figure 2.8: False alarm caused by the camera movement

2.4.3 Template matching

The template matching algorithm uses a template and slides the template over the search area of the source image and measures the similarity values as introduced in Section 1.2.2. This algorithm can find the target well if the template is exact; however, the target cannot be detected when the orientation or the size of the target in the source image are different with the template. Also, the template matching algorithm is computationally expensive since it slides the template over the entire source image at each time-step.

Figure 2.9: Template matching

Figure 2.10: False alarm occurred by a wrong template

2.4.4 Edge detection

The edge detection algorithm firstly determines a modified, bi-directional derivative of the grayscale point values and the pixels which are smaller than the threshold values are removed. By doing that, a binary image is obtained which shows only high contrast values as 1 and these indicate edges of the image.

Figure 2.11: Edge detection and extracted pentagons

The edge detection algorithm is also affected by the background of the image. For example, if there are some clouds on the background, the algorithm finds even the edge of the clouds so that generates some false alarms. However, the background problem can be resolved with the region of interest and this algorithm is relatively computationally cheaper than other algorithms.

2.5 Feature points extraction using edge detection

The color detection algorithm has frequent false alarms and is affected by the light conditions. Also, the optical flow algorithm is not an effective method since the camera is moving. The template matching algorithm shows good results and it is not affected by the light

Figure 2.12: False detection by the edge of the clouds (left) feature points occlusion (right)

Table 2.1: Comparison of computer vision algorithms

Algorithms	**Advantages**	**Disadvantages**
Color detection	Low computation cost Shows an accurate contour	Sensitive to the light condition Confusion with the background
Optic flow	Detection of moving object	High computation cost Shows a vague contour Sensitive to camera movement
Template matching	Accurate detection False alarm rate is low	High computation cost Needs many templates
Edge detection	Low computation cost Shows an accurate contour	Confusion with the background

conditions; however, many templates are needed to detect the right target and a number of templates make the computation expensive. Therefore, the edge detection is the most adequate computer vision algorithm among these algorithms implemented to detect the feature points of the threat aircraft because this algorithm shows the contours of the threat aircraft well which are needed for the extraction of feature points and is computationally cheaper.

After the detection process, feature points of the threat aircraft must be extracted from the aircraft image identified using the edge detection algorithm. Within a given image containing a threat aircraft, the image is cropped to the region of interest, converted to grayscale, and then processed using the Canny Edge Detection algorithm [31]. The purpose

of the edge detection algorithm is to determine a modified, bi-directional derivative of the grayscale point values and to then remove all pixels that fall below a threshold value. This process leaves a binary image in which points with a value of 1 represent high contrast gradients (i.e., edges).

Edges detected using the preceding method are assumed to be either the exterior edges of the aircraft image or aircraft contours contained within the polygon that is defined by these exterior edges. Next, the largest pentagon containing the identified edges [32, 33] are found and the vertices of this pentagon are associated with the feature points of the threat aircraft. Because the geometry of the threat aircraft is assumed to be known, the pixel distance between these feature points can be used to estimate the distance from the camera to the threat aircraft.

Therefore, the algorithm finds the feature points of the threat aircraft using the edge detection algorithm as shown in Figure 2.11. These feature points are used for the pose estimation process to compute the orientation of the threat aircraft and it is described in the next chapter.

2.6 Summary

In this chapter, several computer vision algorithms for the threat aircraft detection and the feature points extraction are introduced and compared. The edge detection algorithm is the most adequate one for this process since the edge detection algorithm has the low computation cost which is important for sense and avoid systems and the false alarms can be resolved easier than the other algorithms. The computer vision algorithm is applied to the image data from the ground camera system and the algorithm gives the feature points of the threat aircraft successfully.

Chapter 3

Pose Estimation Algorithm

3.1 Introduction

A beneficial point of the camera sensor for a sense and avoid technology is that it can see the orientation of the threat aircraft, while other sensors only provide the range and/or bearing. In order to compute the orientation of the threat aircraft, the pose estimation process is needed. After obtaining the feature points of the threat aircraft in the previous process, the feature points are used for the pose estimation process. The POSIT algorithm [34] is used for the pose estimation process of this research and the rotation matrix R_{BI} that maps free vectors from the inertial reference frame to the threat aircraft body frame is obtained as a result of this process.

3.2 Reference frames

For this research, three reference frames are used: the inertial reference frame, camera-fixed reference frame and aircraft body-fixed reference frame. For the inertial reference frame, North east down (NED) frame is used and x_i, y_i and z_i indicate north, east and down direction, respectively.

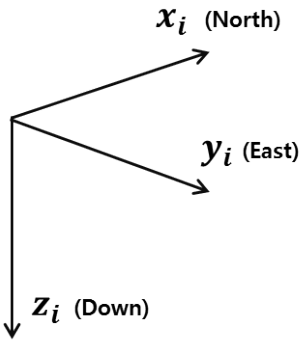


Figure 3.1: Inertial reference frame (North east down (NED) frame)

In the camera-fixed reference frame, the z_c direction indicates the boresight direction of the lens. The x_c and y_c indicates the vertical and horizontal coordinates for the image, respectively.

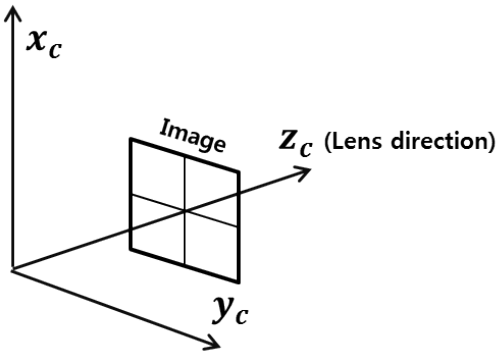


Figure 3.2: Camera-fixed reference frame

The aircraft body-fixed reference frame uses the threat aircraft body as a reference. Therefore, x_b and y_b directions are the nose of the aircraft and the right wing direction, respectively. Also, the z_b direction becomes the down direction of the aircraft body.

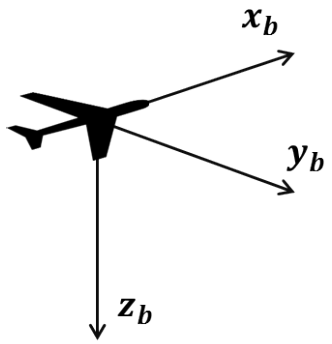


Figure 3.3: body-fixed reference frame

These three frames indicate each reference axis and free vectors in each frame can be transformed to vectors in another reference frame.

3.3 Rotation matrices

As described earlier, the free vectors in each frame can be transformed to vectors in another frame. To do this transformation, rotation matrices are needed [35]. For example, the rotation matrix about the x-axis with ϕ degree is described by:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{3.1}$$

Also, the rotation matrices about the y-axis with θ and z-axis with ψ are:

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.2}$$

$$R_z(\psi) = \begin{bmatrix} \cos\phi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

For the pose estimation process, a rotation matrix in three dimensions is needed to obtain the transformation matrix from a frame to another. Therefore, a rotation matrix which rotates about z-y-x axis-order is computed using matrix multiplication as:

$$\begin{aligned} R_{zxy} &= R_x(\phi)R_y(\theta)R_z(\psi) \\ &= \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \end{aligned} \tag{3.4}$$

From this rotation matrix, Euler angles can be extracted [36]. In order to do this, the R_{zyx} is redefined as:

$$R_{zyx} = \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix} \tag{3.5}$$

and ϕ can be obtained by:

$$\phi = \tan^{-1} \frac{r_{12}}{r_{22}} \tag{3.6}$$

Also, $\cos\theta$ is computed by:

$$\cos\theta = \sqrt{r_{00}^2 + r_{01}^2} \tag{3.7}$$

Therefore, θ is:

$$\theta = \tan^{-1} \frac{-r_{02}}{\cos\theta} \tag{3.8}$$

Finally, ψ becomes:

$$\psi = \tan^{-1}(\frac{r_{20}\sin\phi - r_{10}\cos\phi}{r_{11}\cos\phi - r_{21}\sin\phi}) \tag{3.9}$$

In this way, the rotation matrix can be obtained from three Euler angles and Euler angles can be computed from a rotation matrix.

3.4 Pose estimation process

For the pose estimation process, the author assumes the five feature points are visible in the image and the feature points are detected by the edge detection algorithm. Also, the geometry of the threat aircraft is known. With these assumption, the pose estimation process is implemented using the POSIT algorithm [34], which determines R_{BC}, the rotation matrix relating a camera-fixed reference frame to the aircraft body-fixed reference frame. The

purpose of the pose estimation process is to compute the orientation of the threat aircraft on the image in the inertial reference frame with the known camera orientation. Therefore, it is assumed that the rotation matrix R_{CI} that maps free vectors from the inertial reference frame to the camera-fixed reference frame is known, based on the orientation data of the camera. Finally, the rotation matrix R_{BI} that maps free vectors from the threat aircraft body-fixed frame to the inertial reference frame is obtained using matrix multiplication as:

$$R_{BI} = R_{BC}R_{CI} \tag{3.10}$$

From R_{BI}, the Euler angles can be extracted using (3.5)-(3.9) and these Euler angles are the roll, pitch and yaw angles of the threat aircraft, respectively. Therefore, the orientation of the threat aircraft can be computed with this pose estimation process and the known camera orientation.

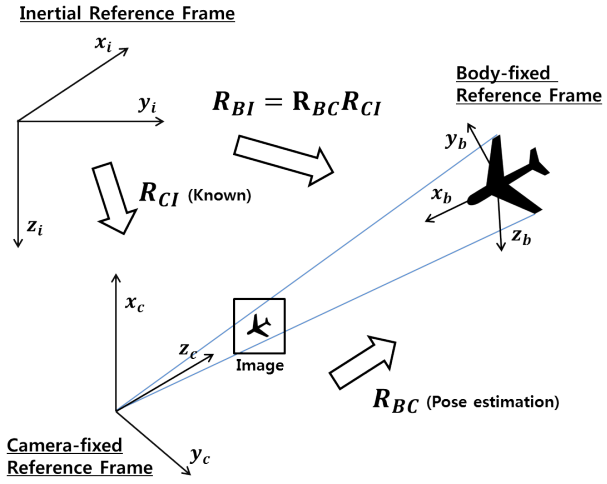


Figure 3.4: Pose estimation geometry

Table 3.1: Average error of pose estimation experiments

Value	**Roll**	**Pitch**	**Yaw**
Average Error Magnitude (°)	3.4	2.7	11.6

3.5 Pose estimation experiments results

The pose estimation experiment was conducted using video images obtained from the ground camera system and the results are compared with the orientation data obtained by the eS-PAARO avionics. Table 3.1 shows the average error magnitude between the orientation estimates from the pose estimation algorithm and direct measurements from the onboard avionics system which is obtained from the flight tests. The pose estimation algorithm provides reasonably accurate estimates of the threat aircraft attitude, expressed using conventional Euler angles. One source of error is the synchronization error between the camera imagery and the camera-mounted PixHawk. Also, the fact that the camera was mounted on the ground resulted in an image set that was less rich than would be obtained using air-to-air imagery. A flight campaign planned for Fall 2017 will obtain a richer image data set.

Figure 3.5: Pose estimation

3.6 Summary

In this chapter, the pose estimation algorithm to obtain the orientation of the threat aircraft in the image is described. An experiment with a ground camera system is also implemented. The experimental results show that the pose estimation algorithm can compute the orientation of the threat aircraft using only the camera image and the camera orientation when the aircraft is visible in the image.

Chapter 4

Aircraft Trajectory Prediction Algorithm

4.1 Introduction

As described in Section 1.1, most sensors other than the camera sensor can only provide the range and/or bearing of the threat. Using this position data of the threat, the future trajectory of the threat aircraft can be computed. If a threat aircraft flies along a straight path, it is not hard to predict its future trajectory using only the position data. If the threat aircraft begins to turn, however, then trajectory predictions based solely on position will accrue error; if the error is sufficiently large, it could compromise the prediction algorithm's ability to inform an avoidance decision. This chapter shows that the additional information obtained by estimating the threat aircraft orientation enables a more accurate prediction of the threat aircraft trajectory. The previous chapter presented an algorithm that estimates the pose of the threat aircraft. In this chapter, an algorithm to predict the trajectory of the threat aircraft over a fixed time horizon is presented, using the roll angle from the

pose estimate and a velocity estimate of the threat aircraft. Finally, the prediction method compares favorably with a more conventional method that uses only position data.

4.2 Aircraft motion model

Having obtained the position and orientation of the threat aircraft, one may estimate the turn rate using the airspeed and roll angle. In this effort, constant-altitude flight is assumed and a two-dimensional coordinated turn (CT) model [37] is used. Although the CT model assumes constant speed and turn rate, these parameters using vision-based estimates are updated under the assumption that the threat aircraft adjusts its turn rate in a quasi-steady manner.

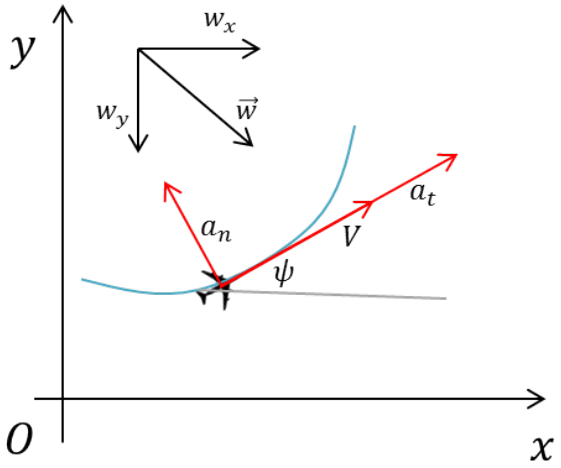


Figure 4.1: Aircraft motion model

Treating the threat as a dynamic particle moving in the horizontal plane, the motion model is:

$$\dot{x}(t) = V\cos\psi(t) + w_x \tag{4.1}$$

$$\dot{y}(t) = V\sin\psi(t) + w_y \tag{4.2}$$

$$\dot{V} = a_{\mathrm{t}}(t) \tag{4.3}$$

$$\dot{\psi}(t) = a_{\mathrm{n}}(t)/V \tag{4.4}$$

where $x, y, \psi, v, a_{\mathrm{t}}, a_{\mathrm{n}}, w_x$ and w_y represent the position of the aircraft, heading angle, speed, tangential and normal acceleration and (constant) wind disturbances, respectively. This model is illustrated in Figure 4.1. In this model, the aircraft speed is constant so that a_{t} is zero and a_{n} is a constant. Using (4.1)-(4.4), noting the assumption that $\omega = \dot{\psi}$ remains constant, and defining the state vector $X = [x, \dot{x}, y, \dot{y}]$, the following motion model is obtained:

$$\begin{aligned}
\dot{X}(t) &= \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \\ \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ \frac{d}{dt}(v(t)\cos\psi(t) + w_x) \\ \dot{y}(t) \\ \frac{d}{dt}(v(t)\sin\psi(t) + w_y) \end{bmatrix} \\
&= \begin{bmatrix} \dot{x}(t) \\ -\omega(\dot{y}(t) - w_y) \\ \dot{y}(t) \\ \omega(\dot{x}(t) - w_x) \end{bmatrix} = A(\omega)X(t) + W(\omega) \qquad (4.5) \\
A(\omega) &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & 0 & 1 \\ 0 & \omega & 0 & 0 \end{bmatrix}, W(\omega) = \begin{bmatrix} 0 \\ \omega w_y \\ 0 \\ -\omega w_x \end{bmatrix}
\end{aligned}$$

The slowly varying turn rate ω is an important parameter in the model (4.5) and two approaches to estimate its value are considered. In the first approach, ω is inferred from the

recent position history. In the second, the airspeed and roll angle are estimated in order to compute the turn rate from the CT model. Details are given in Sections 4.4 and 4.3.

4.3 Position only (PO) approach

A particle dynamic model, or "performance" model, is often used for aircraft trajectory design and analysis. In this model, the aircraft is a particle subject to external forces such as thrust, drag, lift, and weight. The system state is defined by the particle's position

Figure 4.2: The PO approach geometry

and speed. Assuming constant-altitude and constant-speed motion, the turn rate at the k^{th} time-step can be inferred from the rate of change of the system state, i.e., the speed V and acceleration $\vec{a_k}$, as follows:

$$\omega_k = \frac{\|\vec{a_k}\|}{V} \tag{4.6}$$

The resulting turn rate can then be used to predict the trajectory over some time horizon using (4.5).

4.4 Position-plus-roll-angle body (PPR) approach

Alternatively, the turn rate can be computed using the roll angle of the threat aircraft, as obtained from pose estimation. For a coordinated turn, there is a simple relationship

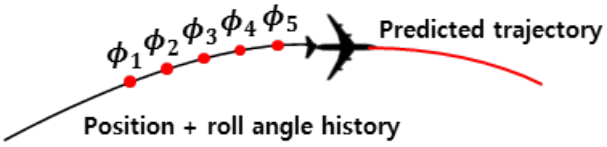


Figure 4.3: The PPR approach geometry

between the turn rate ω and the roll angle of an aircraft ϕ, assuming a constant altitude and speed [38]. Because the lift force is deflected through the roll angle, the lift L must increase in order for the vertical component of lift to balance the weight W:

$$L\cos\phi = W \tag{4.7}$$

The lateral component of the increased lift force establishes the centripetal acceleration required for the turn. The load factor $n_k = L/W$ at the k^{th} time-step is computed as:

$$n_k = \frac{L}{W} = \frac{1}{\cos\phi_k} \tag{4.8}$$

Finally, the turn rate is

$$\omega_k = \frac{g\sqrt{n_k^2 - 1}}{V} \tag{4.9}$$

where g and V are the acceleration of gravity and the aircraft speed. This computed turn rate is used in (4.5) to predict the future trajectory of the threat.

4.5 Simulation setup

To assess the proposed prediction algorithm more thoroughly, two simulations are constructed: one for the PO approach and another for PPR approach. The turn rate is a critical parameter for both models. In the former case, it is determined from the position

history. In the latter, is determined using the estimated roll angle. Simulations were performed with and without the addition of zero-mean, Gaussian white noise to the position and roll angle. Since the threat aircraft is assumed to be known, there is no additional loss of generality in assuming a known turn rate limit. Therefore, the magnitude of the roll angle limitation is assumed to be 45°, corresponding to the maximum load factor at the given speed, and the magnitude of the roll *rate* limitation is also assumed to be 10°/s. The roll angle histories for two simulation cases are shown in Figure 4.8.

For the PO approach, the acceleration and velocity are estimated at each time-step using the previous 5 consecutive positions. The turn rate is computed from these values using (4.10) at each time-step and substituting into (4.5). For the PPR approach, the turn rate used in (4.5) is estimated from (4.9).

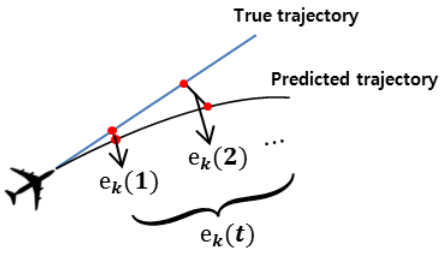


Figure 4.4: Instantaneous distance error $e_k(t)$

At each time-step, for each approach, the instantaneous distance error between the predicted trajectory and the true trajectory is computed over a 10 second horizon at k^{th} time-step and this instantaneous distance error function is defined as $e_k(t)$ as shown in Figure 4.4 and 4.5. Therefore, the higher increasing rate of $e_k(t)$ indicates the bigger separation between the predicted trajectory and the true trajectory. To approximate the increasing rate of $e_k(t)$, a function which fits into this $e_k(t)$ is found using a fitting test with various functions. The Figure 4.5 shows the results of the fitting test and it indicates the 2^{nd} degree

polynomial regression finds the better graph than the other functions. Therefore, the $e_k(t)$ can be defined with the 2$^{\text{nd}}$ degree polynomial regression as:

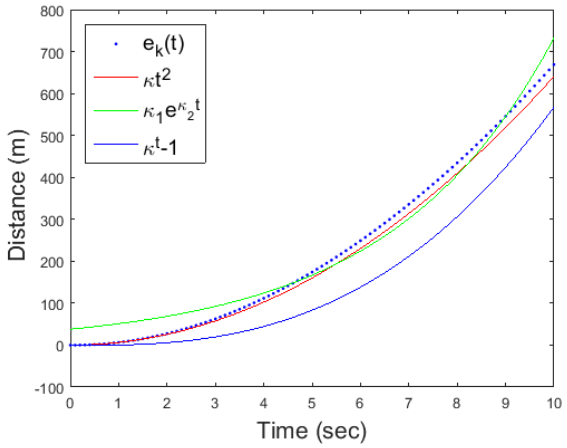


Figure 4.5: $e_k(t)$ graph and the fitting test results

$$e_k(t) = \kappa_1 t^2 + \kappa_2 t + \kappa_3 \tag{4.10}$$

where κ_1, κ_2 and κ_3 indicate constants of the equation. In order to assess and compare the separation of two approaches, κ_1 is used as a separation constant κ since κ_1 and κ_2 can generally be neglected and $\kappa_1 t^2$ shows a reasonable fitting results as shown in Figure 4.6.

This separation constant κ is computed for both approaches at each time-step to compare the prediction performance of two approaches and the results are described in the next section.

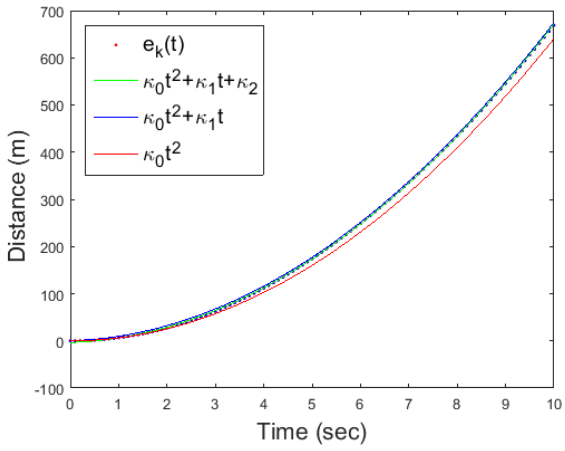


Figure 4.6: The fitting comparison among graphs with 2nd polynomial regression

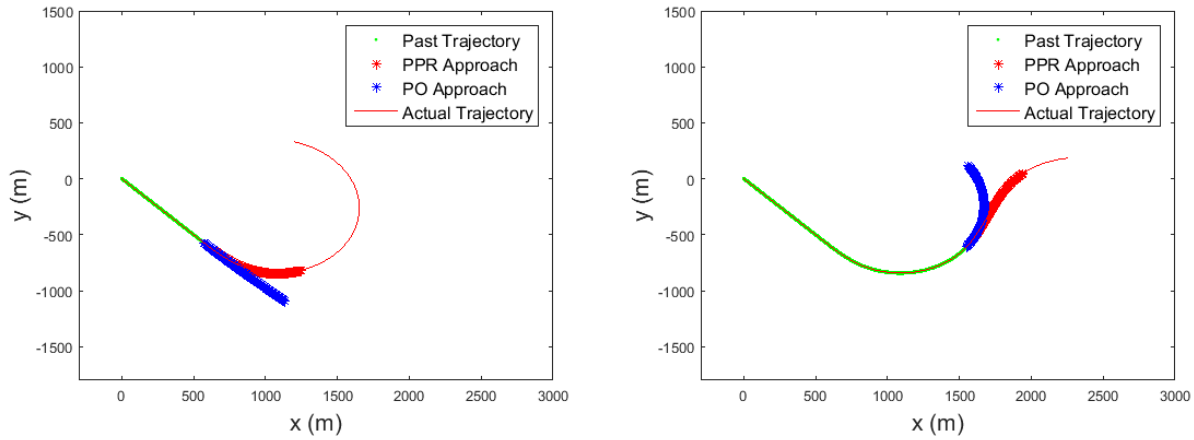


Figure 4.7: Simulation examples

4.6 Simulation results

Figure 4.9 compares the PO and PPR prediction methods for two simulation cases. Note that κ increases just before time $t = 10$ seconds, when the aircraft begins to turn. When the threat aircraft begins rolling at 10 seconds, both prediction algorithms react to the changing turn rate and correct the prediction so that κ decreases. The time taken to correct the predicted trajectory is defined as "prediction correction time".

The "prediction correction time" in Table 4.2 indicates that κ in the PPR approach begins

Table 4.1: Simulation parameters

Parameter	**Values**
Speed (m/s)	100
Turn rate change rate (°/sec)	10
Prediction range time (sec)	10
Total simulation time (sec)	30
Data frequency (Hz)	10
Position noise covariance	0.5
Roll angle noise covariance	3
Wind speed (m/s)	2 - 4
Wind direction (°)	0 - 360

to recover, on average, 2.4 seconds faster than the PO approach. For small UAS flying in close proximity, the additional lead time provided by the PPR approach could be critical in making an avoidance decision. For the simulations that include noise, this recovery time is more difficult to define and compute, but a moving average, as shown in Figure 4.10, suggests the PPR approach again recovers to the true trajectory faster than the PO case. Table 4.2 shows the average value of κ for 576 simulations involving various maneuvers in various wind conditions. Note that average magnitude of κ is lower for the PPR approach than for the PO approach, particularly in the case of noise where the PO approach suffers significantly, as shown in Table 4.1. This result is consistent with the analysis of Section 4.7, which shows that the PO approach is more sensitive to noise than the PPR approach.

Table 4.2: Trajectory recovery time and κ of two approaches

Parameter	**PPR**	**PO**
Prediction correction time (sec)	4.7	7.1
κ without Noise	1.5	1.6
κ with Noise	1.7	3.3

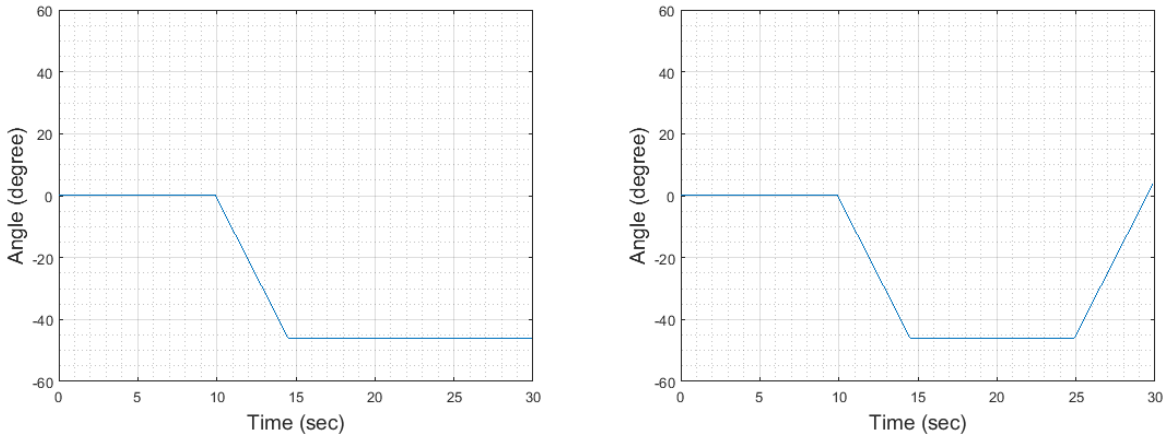


Figure 4.8: Roll angle history for simulations

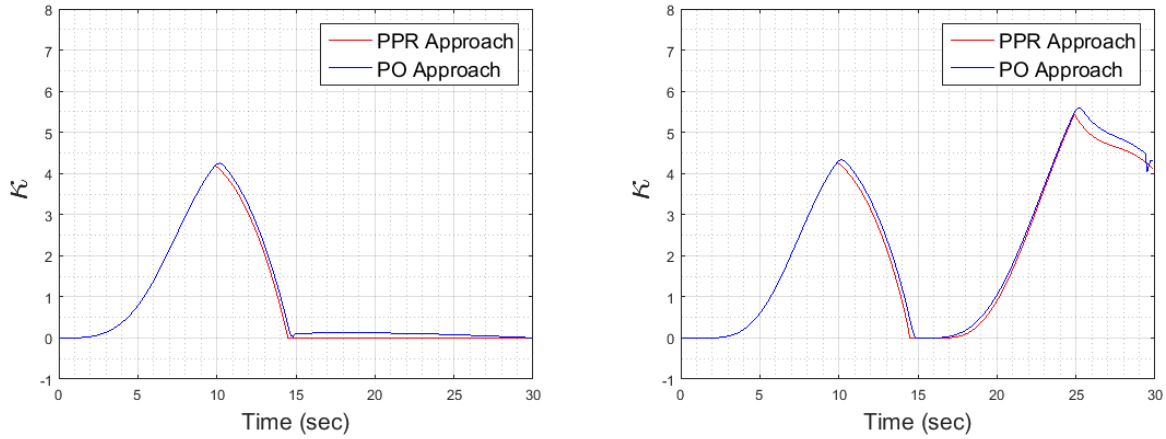


Figure 4.9: κ-time graphs without noise

4.7 Sensitivity to error in variables

In the previous section, the turn rate of the threat aircraft is computed using two different approaches. These approaches are subject to error in the position and roll angle estimates. This section analyzes this sensitivity to estimation error. For the PO approach, the turn rate is computed using (4.10). The error $\delta\omega_{PO}$ in the turn rate estimate due to error in the

position is:

$$\begin{aligned}\delta\omega_{PO} &= \frac{\partial}{\partial a_x}\left[\frac{\sqrt{a_x^2+a_y^2}}{\sqrt{v_x^2+v_y^2}}\right]\delta a_x + \frac{\partial}{\partial a_y}\left[\frac{\sqrt{a_x^2+a_y^2}}{\sqrt{v_x^2+v_y^2}}\right]\delta a_y \\ &+ \frac{\partial}{\partial v_x}\left[\frac{\sqrt{a_x^2+a_y^2}}{\sqrt{v_x^2+v_y^2}}\right]\delta v_x + \frac{\partial}{\partial v_y}\left[\frac{\sqrt{a_x^2+a_y^2}}{\sqrt{v_x^2+v_y^2}}\right]\delta v_y \\ &= \frac{1}{\sqrt{a_x^2+a_y^2}\sqrt{v_x^2+v_y^2}}(a_x\delta a_x + a_y\delta a_y) \\ &- \frac{\sqrt{a_x^2+a_y^2}}{(v_x^2+v_y^2)\sqrt{v_x^2+v_y^2}}(v_x\delta v_x + v_y\delta v_y)\end{aligned} \tag{4.11}$$

For the PPR approach, with turn rate estimated as in (4.9), the error in turn rate due to error in the roll angle estimate and the position is

$$\begin{aligned}\delta\omega_{PPR} &= \frac{\partial}{\partial\phi}\left[\frac{g\sqrt{1/\cos^2\phi-1}}{\sqrt{v_x^2+v_y^2}}\right]\delta\phi \\ &+ \frac{\partial}{\partial v_x}\left[\frac{g\sqrt{1/\cos^2\phi-1}}{\sqrt{v_x^2+v_y^2}}\right]\delta v_x \\ &+ \frac{\partial}{\partial v_y}\left[\frac{g\sqrt{1/\cos^2\phi-1}}{\sqrt{v_x^2+v_y^2}}\right]\delta v_y \\ &= \frac{g}{2V}\frac{\tan\phi}{\cos^2\phi\sqrt{1/\cos^2\phi-1}}\delta\phi \\ &- \frac{g\sqrt{1/\cos^2\phi-1}}{(v_x^2+v_y^2)\sqrt{v_x^2+v_y^2}}(v_x\delta v_x + v_y\delta v_y)\end{aligned} \tag{4.12}$$

As seen in (4.11), the turn rate equation of the PO approach ω_{PO} requires $\delta a_x, \delta a_y, \delta v_x$ and δv_y. Also, the turn rate equation for the PPR approach ω_{PPR} depends on $\delta\phi$, δv_x and δv_y. Therefore, ω_{PO} is affected by more variables than ω_{PPR} so that the PO approach is more sensitive to error. As shown in Table 4.3, the turn rate estimated using the PO approach has higher error magnitude, which results in greater trajectory prediction error.

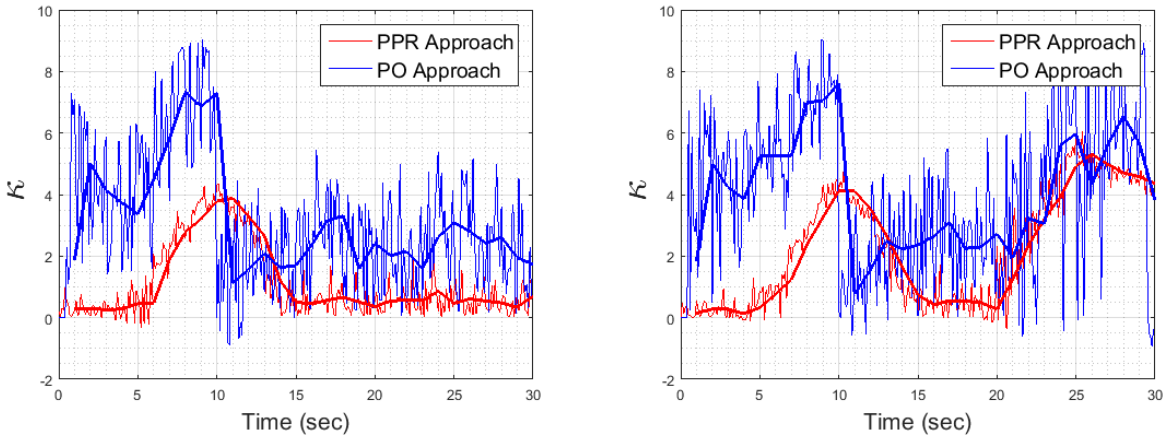


Figure 4.10: κ-time graphs with noise

Table 4.3: Average turn rate error magnitude

Parameter	**PPR**	**PO**
Average Turn Rate Error Magnitude (°/sec)	0.9	4.8

4.8 Trajectory prediction experiments

The prediction algorithm is also applied to the trajectory data obtained in flight tests [12]. In all, sixteen segments of a sample trajectory which is obtained from flight tests were considered (Figure 4.11) and the algorithms described above were applied to each segment. Figure 4.11 shows two of sixteen segments and the prediction algorithm is applied to each path. The red curve represents the predicted trajectory based on the PPR approach and the other is a predicted trajectory based on the PO approach. Also, a blue thin line is the actual path of the data. In this figure, the PPR approach based prediction is closer to the actual data than the PO approach. The κ-time graphs in Figure 4.12 also indicate the PPR approach provides a better prediction performance since κ of the PPR approach is lower than the PO approach for most of the time. Table 4.4 compares κ for the two prediction methods and the κ of two approaches is not as low as it was for the simulation results. It is important to know that

the eSPAARO did not fly with a truly constant speed nor at a truly constant altitude, as assumed in the CT model. Still, the PPR approach results in lower κ than the PO approach, suggesting that trajectory prediction informed by orientation estimates can provide a more accurate trajectory prediction than one based solely on position measurements.

Table 4.4: Average magnitude of κ for flight tests

Parameter	**PPR**	**PO**
Average κ	7.3	8.7

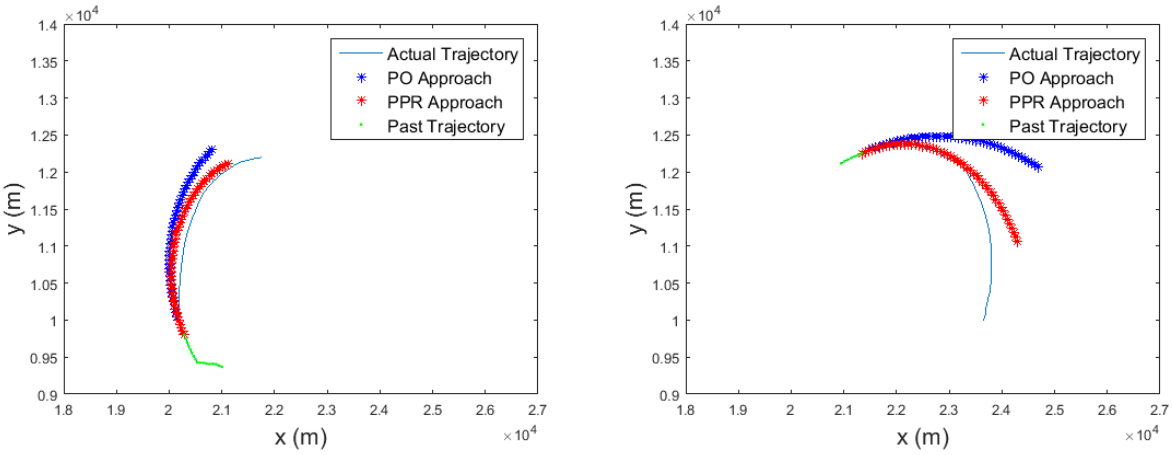


Figure 4.11: Extracted experiment trajectories

4.9 Summary

An algorithm is presented for predicting the trajectory of a fixed-wing aircraft using estimates of the aircraft orientation obtained from a monocular camera. The prediction method incorporates the orientation data into a coordinated turn model for the aircraft motion to predict the trajectory. Prediction results compared favorably with results based solely on the position history, both in accuracy and in speed of response to threat aircraft maneuvers.

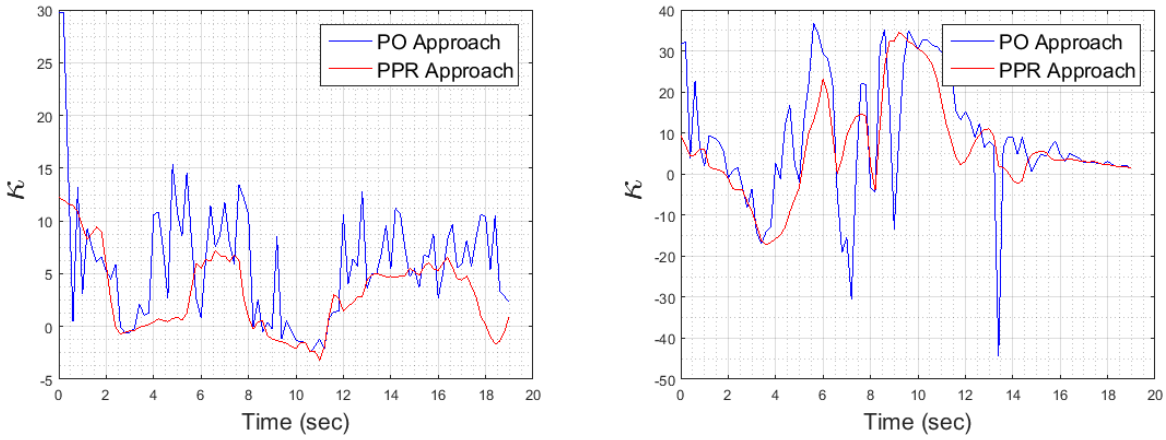


Figure 4.12: κ-time graphs of extracted trajectories

The PPR approach also appears to be less sensitive to noise, both in simulation results and in error sensitivity analysis.

Chapter 5

Aircraft Avoidance Algorithm

5.1 Introduction

In the simulation of the previous chapter, the PPR approach shows quicker response to a trajectory change and is more accurate than the PO approach. If the trajectory of the threat aircraft can be known earlier and more accurately, the host aircraft can react to the threat aircraft earlier. Therefore, the faster and more accurate trajectory prediction of the threat aircraft is important for sense and avoid systems. In this chapter, collision avoidance performances using trajectory predictions generated by the PPR approach and the PO approach are compared and analyzed.

5.2 Avoidance algorithm

The avoidance algorithm consists of two phases: the avoidance phase and trajectory recovery phase. When the host aircraft encounters a threat aircraft, it starts to avoid the threat.

However, it is also important to recover its original trajectory right after the avoidance maneuver to continue to do the mission. Therefore, the avoidance algorithm includes the trajectory recovery phase after the avoidance phase.

5.2.1 Avoidance phase

The CT model is using the turn rate of the aircraft as a control. Assuming a flight at a constant speed and constant altitude, the roll angle of the host aircraft becomes a control. In order to avoid the threat aircraft, the future states of the host can be propagated with 61 possible roll angles (from $-30°$ to $30°$) and the possible future host aircraft trajectories are compared with the future trajectories of the threat aircraft. For the predicted trajectories of the threat aircraft, some uncertainty is added to the nominal predicted trajectory since there are some errors in the pose estimation process as shown in the table 3.1. Therefore,

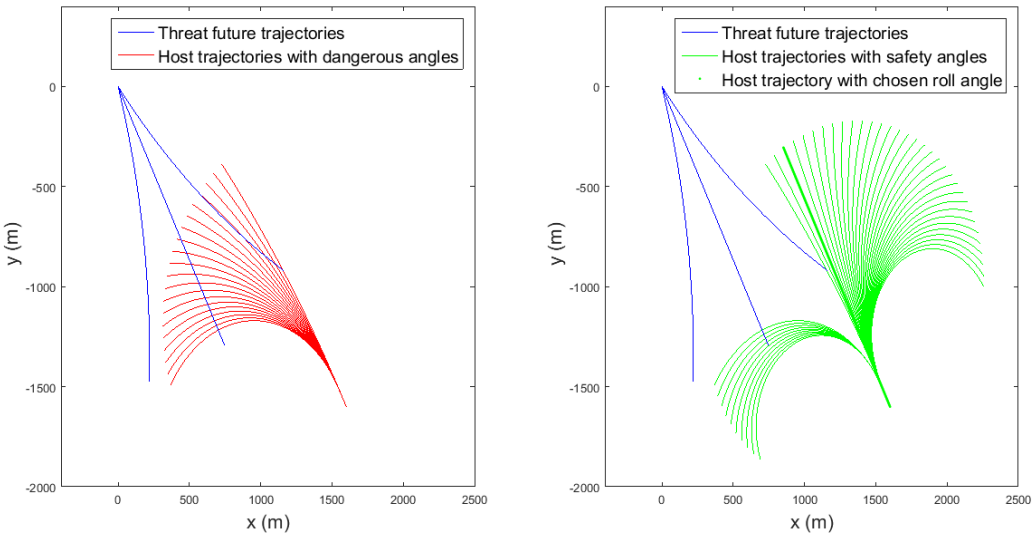


Figure 5.1: Dangerous angle range (left) and safety angle range (right)

the minimum, nominal and maximum predicted trajectories of the the threat aircraft are computed with the yaw angle estimate error of $11.6°$ and the roll angle estimate error of $3.4°$

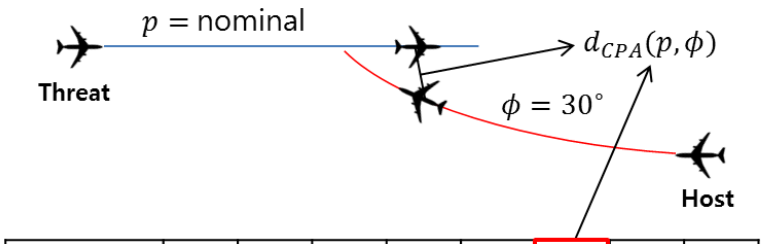
for the PPR approach. For the PO approach, the position error of $50cm$ is added as well. Each trajectory is an array of the future positions over a 10 second time horizon and the number of time-step of the prediction is defined as T_1. Therefore, a $T_1 \times 1$ array $D(p, \phi)$ which indicates the distance between the threat aircraft and the host aircraft of each encounter can be computed with a possible trajectory range of the threat aircraft P and a possible roll angle control range of the host aircraft Φ:

$$
\begin{aligned}
& D(p,\phi) = \sqrt{(X_p^h - X_\phi^t)^2 + (Y_p^h - Y_\phi^t)^2} \\
\text{where} \quad & X_p^h = [x_{h_1}, x_{h_2}, x_{h_3}, \cdots, x_{h_T}] \\
& Y_p^h = [y_{h_1}, y_{h_2}, y_{h_3}, \cdots, y_{h_T}] \\
& X_\phi^t = [x_{t_1}, x_{t_2}, x_{t_3}, \cdots, x_{t_T}] \\
& Y_\phi^t = [y_{t_1}, y_{t_2}, y_{t_3}, \cdots, y_{t_T}] \\
& p \in P = \{\text{minimum, nominal, maximum}\} \\
& \phi \in \Phi = \{-30^\circ, -29^\circ, -28^\circ, \cdots, 29^\circ, 30^\circ\}
\end{aligned}
\tag{5.1}
$$

where X_p^h, Y_p^h, X_ϕ^t and Y_ϕ^t are $T_1 \times 1$ arrays of x, y coordinates of the threat aircraft and the host aircraft at each time-step, respectively. From $D(p, \phi)$, the distance between closest points of approach (CPA) of each encounter are determined by:

$$
d_{CPA}(p,\phi) = min(D(p,\phi)) \tag{5.2}
$$

For this research, a collision is defined with a safety radius R_d ($500ft$ =154.2m), which means if the $d_{CPA}(p, \phi)$ is smaller than the safety radius, it is defined as a collision and the roll angle control ϕ of the encounter is determined as a dangerous angle.



Time(sec)	0	1	2	···	7	8	9	10
$D(p,\phi)$ (m)	600	580	560	···	210	200	210	220

Figure 5.2: Avoidance phase geometry

The figure 5.2 shows an example of this process. In this case, p is nominal and ϕ is 30°. The future distance between the two aircraft at each time-step ($D(p,\phi)$) is computed and the CPA is generated after 8 seconds since the $D(p,\phi)$ is the minimum at this time-step. Therefore, $d_{CPA}(p,\phi)$ of this encounter becomes $200m$. This computation is processed for P and Φ and the dangerous roll angle range Φ_d and the safety roll angle range Φ_s are decided by:

$$\begin{aligned} \phi_d \in \Phi_d &= [min\{\phi \mid CPA(p,\phi) < R_d\}, max\{\phi \mid CPA(p,\phi) < R_d\}] \\ \phi_s \in \Phi_s &= \Phi_d^{\complement} \ (= \Phi \backslash \Phi_d) \end{aligned} \tag{5.3}$$

Once the safety roll angle range Φ_s is computed as shown in Figure 5.1, the smallest absolute angle in Φ_s is chosen as the roll angle control at the next time-step ϕ_u, since the smallest roll angle control minimizes the aircraft maneuver.

5.2.2 Trajectory recovery phase

When the host aircraft encounters the threat aircraft, the distance between the host aircraft and the threat aircraft decreases and the distance starts to increase when the host aircraft

successfully avoids the threat aircraft. The algorithm then changes the phase of the algorithm to the trajectory recovery phase. Once the trajectory recovery phase is started, the distance of closest point on the original trajectory to the current position of the host aircraft (CPO) is computed.

$$d_{CPO} = min(\sqrt{(X_o^h - X_c^h)^2 + (Y_o^h - Y_c^h)^2}) \tag{5.4}$$

where X_o^h and Y_o^h are $T_2 \times 1$ arrays of x, y coordinates of the original trajectory of the host aircraft and X_c^h and y_c^h are $T_2 \times 1$ arrays which consists of x and y coordinate of the current position of the host aircraft and T_2 is the number of the time-step of the entire simulation. Also, the x and y coordinates of CPO are defined as x_{CPO} and y_{CPO} and the vector from the current position of the host aircraft to CPO $\vec{p}$ is also defined as:

$$\vec{p} = [x_{CPO} - x_c^h,\ y_{CPO} - y_c^h]^T \tag{5.5}$$

After the avoidance phase, the current heading angle of the host aircraft ψ_c is different with the desired heading angle ψ_d. In order to recover its trajectory, the host aircraft needs to go toward to the original trajectory and the current heading angle ψ_c is also needed to be recovered to ψ_d. Therefore, the algorithm firstly determines ϕ_u to head toward the original trajectory. For this process, the $\vec{p}$ is needed to be rotated with $-\psi_c$. The ψ_c and the rotated vector $\vec{p_r}$ are computed as:

$$\psi_c = \tan^{-1}(\frac{v_y^h}{v_x^h}) \tag{5.6}$$

$$\vec{p_r} = \begin{bmatrix} p_{rx} \\ p_{ry} \end{bmatrix} = \begin{bmatrix} \cos(-\psi_c) & -\sin(-\psi_c) \\ \sin(-\psi_c) & \cos(-\psi_c) \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} \tag{5.7}$$

where v_y^h and v_x^h are the current speed in x, y direction of the host aircraft. If the p_{ry}, which is a y coordinate of $\vec{p_r}$, is positive, the algorithm chooses ϕ_u as the negative minimum roll angle ($-30°$) and ϕ_u will be chosen as the positive maximum roll angle ($30°$) otherwise.

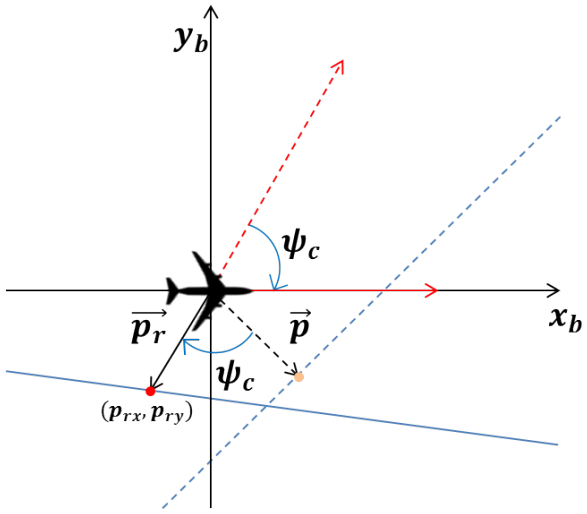


Figure 5.3: Trajectory recovery phase geometry

After the host aircraft becomes close to the original trajectory with the previous process, the current heading angle ψ_c needs to be same with the desired heading angle ψ_d. For this process, a simple P-controller is used. The heading angle is used as a state and the roll angle control ϕ_u is used as a control. Therefore, ϕ_u is computed as:

$$\begin{aligned} \phi_u &= K_p(\psi_d - \psi_c) \\ K_p &= 7 \end{aligned} \tag{5.8}$$

where K_p is a proportional coefficient for the P-controller. In order to find an adequate

K_p, a simple simulation is implemented. Each simulation starts with the heading angle error of 12° and various K_P values are used for each simulation. The Figure 5.4 shows the error convergence simulation results with various K_p and the heading angle error converges fastest when K_p is 7 among 7 values (1.5, 3, 7, 10, 20 and 50).

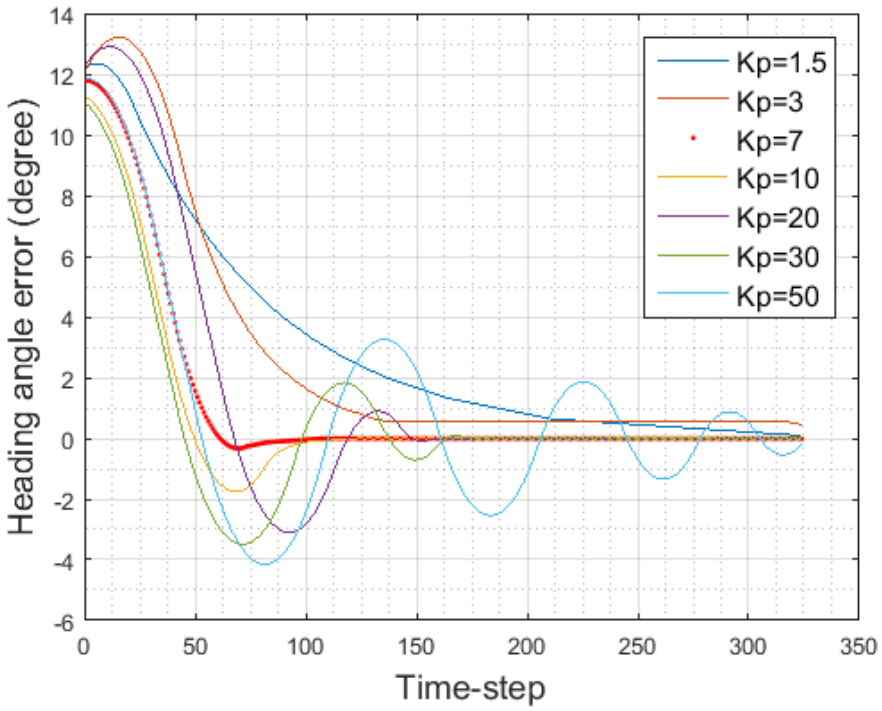


Figure 5.4: Estimation of an adequate K_p value

5.3 Simulation results

To check the avoidance performance based on the PPR approach and the PO approach, a collision avoidance simulation is constructed. As described in the previous section, both the threat aircraft and the host aircraft use the same motion model and a wind model is also applied. The trajectory prediction time is defined as 10 seconds, the total simulation time is 60 seconds and the speed of both aircraft is 100 m/s.

Table 5.1: Avoidance Simulation parameters

Parameter	**Values**
Host aircraft speed (m/s)	100
Threat aircraft speed (m/s)	100
Prediction range time (sec)	10
Total simulation time (sec)	60
Data frequency (Hz)	10
Host roll angle control range $(^\circ)$	-30 $\tilde{3}0$
Wind speed (m/s)	1
Wind direction $(^\circ)$	200
Safe radius (m)	152.4

This simulation firstly shows how the avoidance algorithm works. As shown in Figure 5.5 and 5.6, the host aircraft successfully avoids the threat aircraft and recovers its original trajectory. The results show that this avoidance algorithm is a well combined approach of the worst case approach and the optimized approach that are introduced in Section 1.3.2, since the host aircraft avoids the trajectory range which is based on the sensor uncertainty and the host aircraft chooses an optimal roll angle control to minimize its maneuver.

The results also show that the avoidance based on the PO approach is more dangerous than the one based on the PPR approach. As presented in Section 4.6 and 4.7, the PO approach is more sensitive to error so that the κ is higher than the PPR approach. Therefore, the trajectory prediction of the threat aircraft is less accurate and oscillates more than the PPR approach, which makes the avoidance maneuver unreliable. As shown in Figure 5.5, the d_{CPA} is smaller than R_d with the value of $32.8m$ and $64.9m$, while the d_{CPA} of the PPR approach are $256.7m$ and $236.3m$ as shown in Figure 5.6.

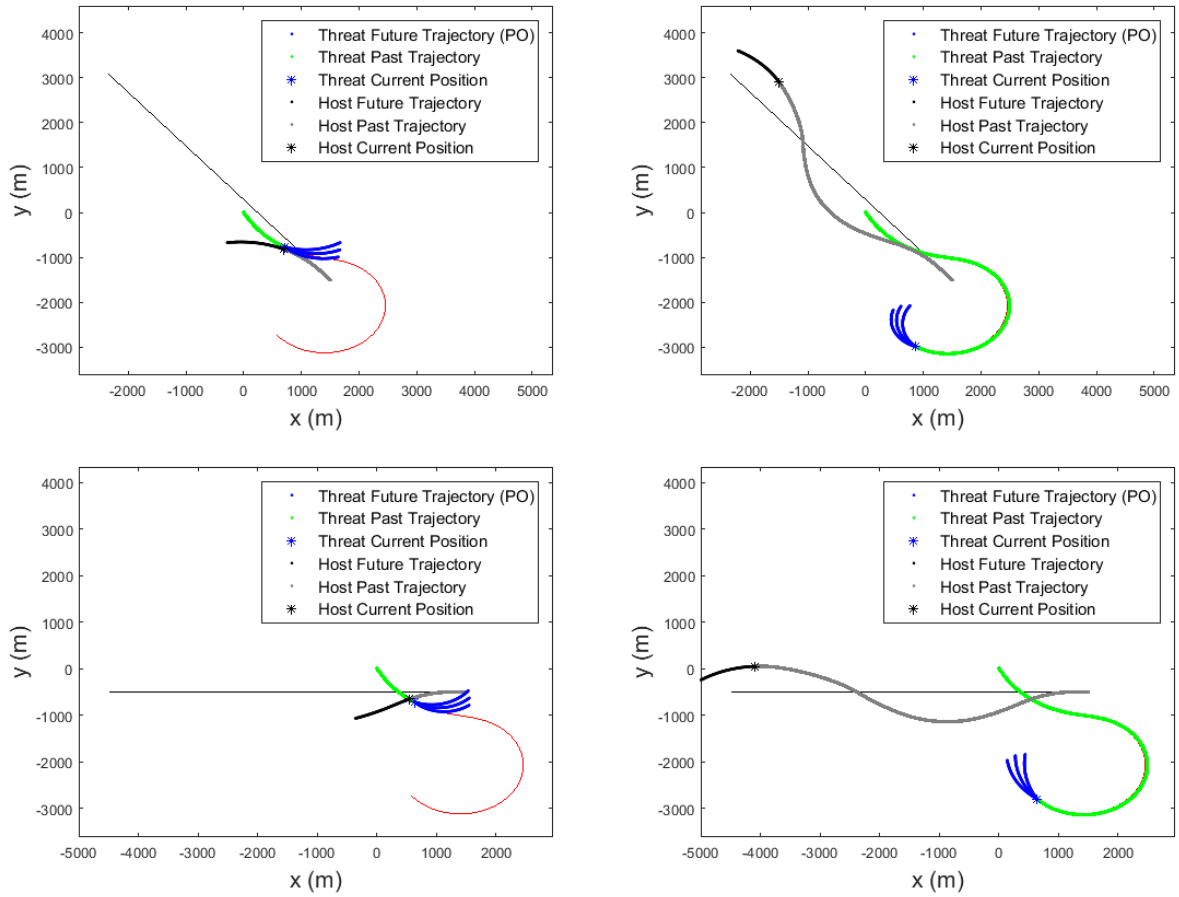


Figure 5.5: Avoidance simulations for the PO approach

5.4 Summary

In this chapter, an avoidance algorithm is introduced. This avoidance algorithm is a combined avoidance algorithm of the worst case avoidance approach and the optimized avoidance approach. The simulation results show that the host aircraft with this avoidance algorithm can avoid the threat aircraft with the least conservative control. Also, the avoidance performance based on the PPR approach and the PO approach are compared. The PPR approach shows a better prediction performance and less sensitivity to sensor errors as described in the previous chapter so that the PPR approach gives the better avoidance performance than the PO approach.

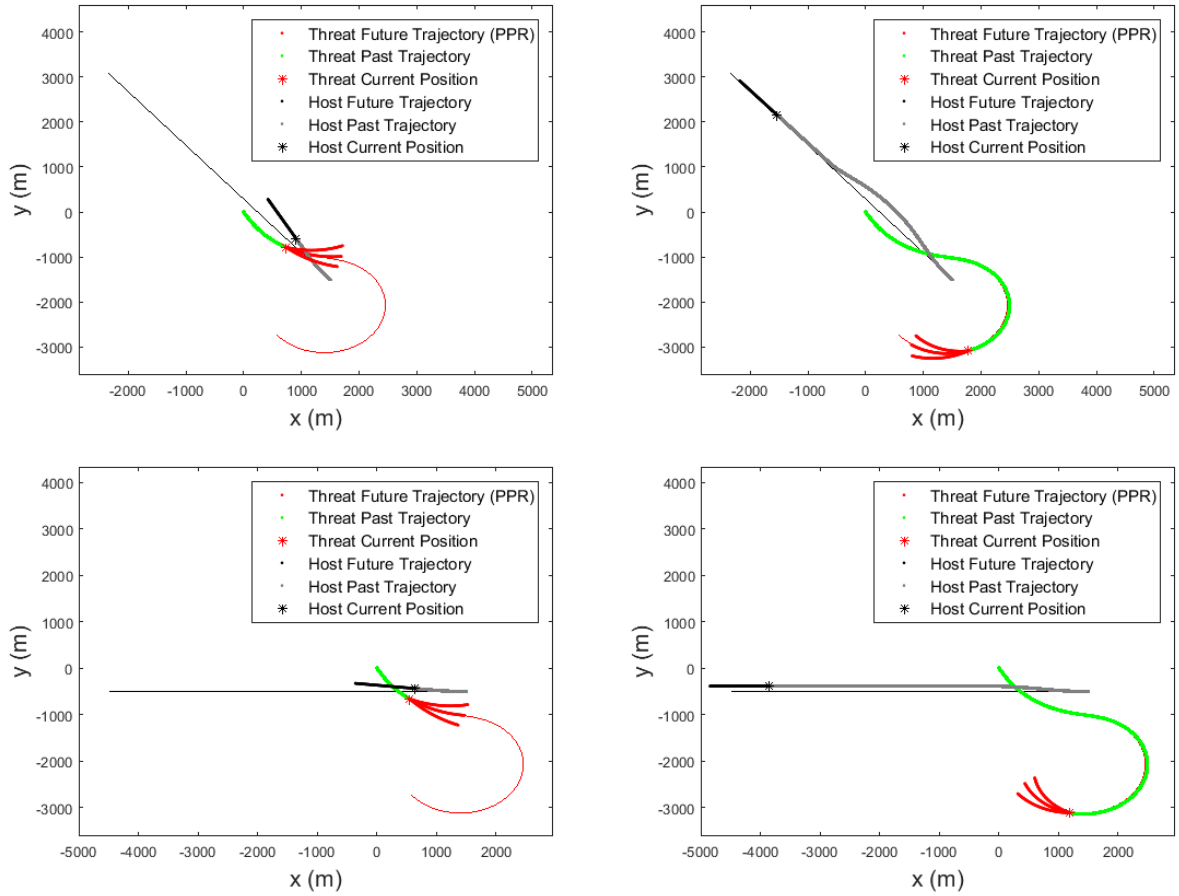


Figure 5.6: Avoidance simulations for the PPR approach

Chapter 6

Conclusion and future work

In this thesis, a trajectory prediction and avoidance algorithm based on computer vision techniques for a small, fixed-wing UAS has been developed. There are a variety of sensors for sense and avoid systems and camera sensors provide the threat orientation data which cannot be obtained from other sensors. Using computer vision techniques and a pose estimation algorithm, the orientation of the threat aircraft in the image was analyzed and the orientation data was used to predict the future trajectory of the threat aircraft and avoid the threat aircraft.

6.1 Computer vision technology

An algorithm has been presented to find the orientation of the threat aircraft using some computer vision techniques and a pose estimation algorithm. Four computer vision algorithms were implemented: color detection, optical flow, template matching and edge detection. Each algorithm had advantages and disadvantages. The Color detection algorithm was not adequate since the threat aircraft does not have a specific color so that there is a high

possibility of false alarms. The optical flow algorithm was a good method to find a moving object in an image. However, it could be applied only when the camera was fixed, hence it was not a proper method for this research. The template matching algorithm was the most accurate method among these four algorithms to find a target in an image when the template was accurate. However, the threat aircraft kept changing its orientation so that a number of templates were needed which made the algorithm computationally expensive. The edge detection algorithm also had some error when the algorithm recognizes even edges of the background. However, this problem could be solved with a region of interest and this algorithm was computationally cheap. Therefore, the edge detection algorithm was chosen for this research to find feature points of the threat aircraft.

6.2 Pose estimation algorithm

Feature points of the threat aircraft were detected by the computer vision algorithm and these feature points were used for the pose estimation process. Three reference frames were used for the pose estimation process: the inertial reference frame, camera-fixed reference frame and aircraft body-fixed reference frame. Using the POSIT algorithm, the rotation matrix which maps free vectors from the camera-fixed reference frame to the body-fixed reference frame was obtained. Therefore, the orientation of the threat aircraft was computed with the known rotation matrix relating the inertial reference frame to the camera-fixed reference frame. An experiment with an UAV and a ground camera system was implemented and the results showed that the pose estimation algorithm could estimate the orientation of the threat aircraft well with the roll angle error of 3.4°, pitch angle error of 2.7° and yaw angle error of 11.6°.

6.3 Aircraft trajectory prediction algorithm

After getting the orientation of the threat aircraft, this orientation data was used to predict the future trajectory of the threat aircraft. A coordinated turn model was used and two approaches were introduced to predict the turn rate and the future trajectory of the threat aircraft: PPR approach and PO approach. The PPR approach used the position data and roll angle data to compute the turn rate of the threat aircraft. On the other hand, the PO approach used only position data to compute the turn rate. The prediction performances of these two approaches were compared using simulations. The prediction performance was assessed with a separation constant κ and the results showed that κ of the PPR approach was lower than the PO approach, which means that the PPR approach provides more accurate prediction than the PO approach. In particular, a significant magnitude difference of κ between the two approaches in the noise simulation indicates that the PO approach is more sensitive to error in variables. Also, two approaches were compared with the real experiment data of an UAV and the experimental results showed that the PPR approach has lower κ than the PO approach.

6.4 Aircraft avoidance algorithm

Based on the predicted trajectory of the threat aircraft, an avoidance algorithm was introduced. The sensor uncertainty was added to the predicted trajectory of the threat aircraft. The closest points of approach between possible trajectories of the threat aircraft and possible trajectories of the host aircraft were computed; and the host aircraft determined the dangerous roll angle range and the safe roll angle range based on the closest points of approach computation. Once the host aircraft successfully avoided the threat aircraft, the

trajectory recovery phase was started. The host aircraft recovered its trajectory based on the current heading angle, initial heading angle and distance to the original trajectory from the current position of the host aircraft. The simulation results suggested the avoidance algorithm can avoid the threat aircraft and recovers the original trajectory. Also, the avoidance performances of the PPR approach and the PO approach were compared. The results showed that the avoidance performance based on the PPR approach is more efficient and safer since the trajectory prediction based on the PO approach is less accurate and oscillates more than the PPR approach.

6.5 Future work

At this point, the vision-based prediction algorithm was designed for a specific use case under restrictive assumptions (e.g., constant speed and altitude) and the results from analysis of flight test data were limited by the richness of the encounter perspectives. Ongoing work aims to refine and improve the algorithm by relaxing these assumptions, obtaining air-to-air imagery and, finally, extending the approach to multi-rotor aircraft. The ultimate aim is to build a complete, real-time system enabling small UAS to see and avoid threats.

Bibliography

[1] Kuchar, J., & Drumm, A. C. (2007). The traffic alert and collision avoidance system. *Lincoln Laboratory Journal, 16*(2), 277.

[2] Department of Transportation, Federal Aviation Administration. (2013). *Pilot's Handbook of Aeronautical Knowledge*.

[3] Yu, X., & Zhang, Y. M. (2015). Sense and avoid technologies with applications to unmanned aircraft systems: Review and prospects. *Progress in Aerospace Sciences, 74*, 152-166.

[4] Zeitlin, A. (2007). *Technology milestones-detect, sense and avoid for unmanned aircraft systems*. Paper presented at the AIAA Infotech@ Aerospace 2007 Conference and Exhibit.

[5] Olson, W. A., & Olszta, J. E. (2010). *TCAS operational performance assessment in the US national airspace*. Paper presented at the Digital Avionics Systems Conference (DASC), 2010 IEEE/AIAA 29th.

[6] Zhao, C., Gu, J., Hu, J., Lyu, Y., & Wang, D. (2016). *Research on cooperative sense and avoid approaches based on ADS-B for unmanned aerial vehicle*. Paper presented at the Guidance, Navigation and Control Conference (CGNCC), 2016 IEEE Chinese.

[7] Sahawneh, L. R., Spencer, J., Beard, R. W., & Warnick, K. (2016). *Minimum required sensing range for UAS sense and avoid systems*. AIAA Infotech@ Aerospace, 1982.

[8] Ramasamy, S., Sabatini, R., Gardi, A., & Liu, J. (2016). LIDAR obstacle warning and avoidance system for unmanned aerial vehicle sense-and-avoid. *Aerospace Science and Technology, 55*, 344-358.

[9] Sapkota, K. R., Roelofsen, S., Rozantsev, A., Lepetit, V., Gillet, D., Fua, P., & Martinoli, A. (2016). *Vision-based Unmanned Aerial Vehicle detection and tracking for sense and avoid systems*. Paper presented at the Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on.

[10] Mcfadyen, A., & Mejias, L. (2016). A survey of autonomous vision-based See and Avoid for Unmanned Aircraft Systems. *Progress in Aerospace Sciences, 80*, 1-17.

[11] Lyu, Y., Pan, Q., Zhao, C., & Hu, J. (2017). *Autonomous Stereo Vision based Collision Avoid System for Small UAV*. AIAA Information Systems-AIAA Infotech@ Aerospace, 1150.

[12] McClelland, H. G., Kang, C., Woolsey, C. A., Roberts, A. K., Buck, D., Cheney, T., & Warnick, K. (2017). *Small Aircraft Flight Encounters Database for UAS Sense and Avoid*. AIAA Information Systems-AIAA Infotech@ Aerospace, 1152.

[13] Sharifi, M., Fathy, M., & Mahmoudi, M. T. (2002). *A classified and comparative study of edge detection algorithms*. Paper presented at the Information Technology: Coding and Computing, 2002. Proceedings. International Conference on.

[14] Maini, R., & Aggarwal, H. (2009). Study and comparison of various image edge detection techniques. *International journal of image processing (IJIP), 3*(1), 1-11.

[15] Kuruppu, G., Manoj, C., Kodituwakku, S., & Pinidiyaarachchi, U. (2013). *Comparison of different template matching algorithms in high speed sports motion tracking*. Paper presented at the Industrial and Information Systems (ICIIS), 2013 8th IEEE International Conference on.

[16] Thota, S. D., Vemulapalli, K. S., Chintalapati, K., & Gudipudi, P. S. S. (2013). Comparison Between The Optical Flow Computational Techniques. *International Journal of Engineering Trends and Technology, 4*(10).

[17] Fortun, D., Bouthemy, P., & Kervrann, C. (2015). Optical flow modeling and computation: a survey. *Computer Vision and Image Understanding, 134*, 1-21.

[18] de Boer, J., & Kalksma, M. (2015). *Choosing between optical flow algorithms for UAV position change measurement*. 12th SC@RUG 2015 proceedings: Student Colloquium 2014-2015. Rijksuniversiteit Groningen. Universiteitsbibliotheek.

[19] Hisham, M., Yaakob, S. N., Raof, R. A., Nazren, A. A., & Embedded, N. W. (2015). *Template Matching using Sum of Squared Difference and Normalized Cross Correlation*. Paper presented at the Research and Development (SCOReD), 2015 IEEE Student Conference on.

[20] Di Stefano, L., & Mattoccia, S. (2003). Fast template matching using bounded partial correlation. *Machine Vision and Applications, 13*(4), 213-221.

[21] Briechle, K., & Hanebeck, U. D. (2001, March). *Template matching using fast normalized cross correlation*. In Aerospace/Defense Sensing, Simulation, and Controls. International Society for Optics and Photonics, 95-102.

[22] Mahmood, A., & Khan, S. (2012). Correlation-coefficient-based fast template matching through partial elimination. *IEEE Transactions on image processing, 21*(4), 2099-2108.

[23] Farnebck, G. (2003). *Two-frame motion estimation based on polynomial expansion*. SCIA 2003: Image Analysis, 363-370.

[24] Lucas, B. D., & Kanade, T. (1981). *An iterative image registration technique with an application to stereo vision*. In: Proceedings of the International Joint Conference on Artificial Intelligence.

[25] Kuchar, J. K., & Yang, L. C. (2000). A review of conflict detection and resolution modeling methods. *IEEE Transactions on intelligent transportation systems, 1*(4), 179-189.

[26] Sahawneh, L. R. (2016). *Airborne Collision Detection and Avoidance for Small UAS Sense and Avoid Systems*. (Doctoral dissertation) Retrieved from http://scholarsarchive.byu.edu/etd/5840.

[27] Albaker, B., & Rahim, N. (2009). *A survey of collision avoidance approaches for unmanned aerial vehicles*. Paper presented at the technical postgraduates (TECHPOS), 2009 international conference for.

[28] Alturbeh, H. (2014). *Collision avoidance systems for UAS operating in civil airspace*. (Doctoral dissertation) Retrieved from http://dspace.lib.cranfield.ac.uk/handle/1826/9295.

[29] G. Bradski. *The OpenCV Library*. Dr. Dobbs Journal of Software Tools, 2000.

[30] Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc.

[31] Zhao, X. M., Wang, W. X., & Wang, L. P. (2011). Parameter optimal determination for canny edge detection. *Imaging Science Journal, 59*(6), 332-341

[32] Persson, A. H., Bondesson, L., & Börlin, N. (2006). Estimation of Polygons and Areas. *Scandinavian Journal Of Statistics, 33*(3), 541-559.

[33] Ratschek, H., Rokne, J., & Yap, C. (2000). Exact and Optimal Convex Hulls in 2D. *International Journal Of Computational Geometry and Applications, 10*(2), 109.

[34] DeMenthon, D. F., & Davis, L. S. (1992). *Model-based object pose in 25 lines of code.* Paper presented at the European conference on computer vision.

[35] Woolsey, C. A. AOE 3134 Lecture Notes. Department of Aerospace and Ocean Engineering, Virginia Tech.

[36] Day, M., & Games, I. (2012). *Extracting Euler angles from a rotation matrix.* Insomniac Games R&D. Retrieved from http://www.insomniacgames.com/mike-day-extracting-euler-angles-from-a-rotation-matrix.

[37] Li, X. R., & Jilkov, V. P. (2003). Survey of maneuvering target tracking. Part I. Dynamic models. *IEEE Transactions on Aerospace and Electronic Systems, 39*(4), 1333-1364.

[38] John, A. D., & Anderson, J. (1989). *Introduction to flight, 3rd edition.* New York, NY:McGraw-Hill, 326-330.