Collision avoidance for UAV using visual detection

Tamás Zsedrovits[†], Ákos Zarándy[*], Bálint Vanek[*], Tamás Péni[*], József Bokor[*], Tamás Roska[†*]

[†]Pázmány Péter Catholic University, The Faculty of Information Technology, Budapest, Hungary,
[*]Computer and Automation Research Institute of the Hungarian Academy of Sciences (MTA-SZTAKI)
Budapest, Hungary

***Abstract*—Unmanned Arial Vehicles (UAVs) require the development of some on-board safety equipments before inheriting the sky. An on-board collision avoidance system is being built by our team. Due to the strict size, weight, power, and costs constraints, visual intruder airplane detection is the only option. This paper introduces our visual airplane detector algorithm, which is designed to be operational in clear and in cloudy situations under regular daylight visual conditions. To be able to implement the algorithm on-board, we have carefully selected topographic operators, which can be efficiently solved on cellular processor arrays.**

***Index Terms*— image processing, aircraft detection and tracking, UAV, vision-based control, topographic operators, cellular processor arrays**

I. Introduction

NOWADAYS the use of Unmanned Aerial Systems (UAS) in military operations is quite common but not in civil tasks. There are many civil tasks where the UAS could be very useful (surveillance tasks, fire-fighting, meteorological observation, telecommunications etc.). One of the key issues that must be resolved to open up the skies for UAS is to be able to coexist safely and effectively with current manned operations in the national and inter-national airspace [1]. This includes the ability to perform Sense and Avoid (SAA) functions at an "equivalent level of safety" (ELOS) to manned aircraft while not negatively impacting the existing infrastructure and manned Traffic Alert and Collision Avoidance System (TCAS) that create today's safe airspace [2], [3]. The UAS collision avoidance system, or SAA system, needs to detect a hazard, determine if a maneuver is required, communicate and execute that maneuver in time to achieve a specified miss distance. A purely camera based SAA system would provide cost and weight advantages against radar based solutions currently under research [4], [5]. Feasibility and technical characteristics of such system are unknown, since solely electro-optical sensor based approach has not yet been demonstrated.

II. See and avoid system

The goal of our research is to develop a complete autonomous flight control system for Unmanned Aerial Vehicles (UAV). This is a closed loop flight control system with the collision avoidance capability based on visual detection of the approaching object (Fig. 1). The organization of the system is as follows. The input images are recorded by the *Camera*. The recorded pictures are transmitted by *Image Acquisition* to *Preprocessing* block by which the pictures are filtered. The next step of the processing is *Detection*. The images are processed by image processing algorithms to detect the approaching objects. *Data Association & Tracking* is responsible for the combination of the orientation and angle of attack data of the approaching object calculated by *Detection* and the own position and inertial data measured by onboard *INS/GPS* (Inertial Navigation System/Global Positioning System).

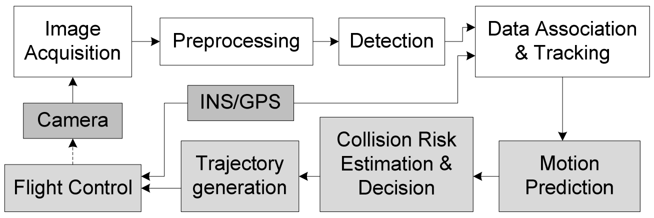


Fig. 1. Diagram of the control system

The second part is the flight control. According to the combined data the relative motion of the approaching object is predicted by *Motion Prediction*. If a risky situation is identified by *Collision Risk Estimation & Decision* a modified trajectory is generated by *Trajectory generation* and the avoiding maneuver is controlled by *Flight Control*. In this paper mainly the image processing part of the currently developed complete autonomous "sense and avoid" flight control system is presented.

A. Coordinate systems

The following three coordinate systems are used in the paper: North-East-Down (NED) - Fixed in one (latitude, longitude) point. We assume flat Earth and the flying distance is short, where axes are defined according in the conventional way [7]. Body - fixed to the c.g. of the aircraft [7]. Camera - a pan, tilt camera is assumed onboard, which is directed in one fixed direction throughout the flight, the camera has offset from the c.g. r', and in the current paper the axes are X_c positive in the direction of West, Y_c positive towards North, and Z_c positive downwards towards the center of the Earth, perpendicular to the $X_c - Y_c$ plane. The rotation matrices between the coordinate systems Body-NED, Body- Camera, Camera-NED are denoted by $L_{bn}(t)$, $L_{bc}(t)$, $L_{cn}(t)$.

B. Measured and estimated variables

We assume there is only one intruder to be detected, so the case of multiple threats is not considered in the paper. The detection of the intruder is formulated as a state estimation problem, where the dynamics are the relative motion of the intruder to our aircraft. The measured output contains all information that can be extracted from the camera images. Since the camera projects the 3D view onto a 2D plane, which is a nonlinear mapping, the measured outputs are nonlinear functions of the states. Even if the motion of the aircrafts are modeled by a linear system, the nonlinearity of the output equation makes it necessary to apply Extended (EKF) or Unscented Kalman Filters (UKF) to estimate the intruder's data [8].

To simplify the filter design the vehicles (intruder and own aircrafts) are modeled in the NED frame by a simple point mass dynamics.

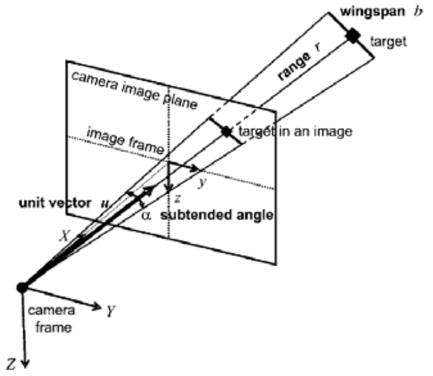


Fig. 2. Subtended Angle Relative State Estimation (SARSE) methods

The relative position of the target, $p_c(t)$, can be expressed in the camera frame as follows:

$$p_c(t) = L_{cb}(t)p(t) = \left[p_{c,x}(t)\; p_{c,y}(t)\; p_{c,z}(t)\right]^T \quad (1)$$

Assuming pinhole camera model the location of the target on the image plane can be computed as follows:

$$z_c(t) = \frac{f}{p_{c,x}(t)}\begin{bmatrix}p_{c,y}(t)\\ p_{c,z}(t)\end{bmatrix} = \begin{bmatrix}p_{m,y}(t)\\ p_{m,z}(t)\end{bmatrix} \quad (2)$$

where f is the focal length of the camera. The details can be seen in Fig. 2. By locating and tracking the intruder on the image plane the image processing unit can determine the direction vector $d(t) = p(t)/\|p(t)\|$ and the subtended angle $\phi(t) = 2\tan^{-1}\left(\frac{b}{2\|p(t)\|}\right)$ under which the target is seen. (The constant b in the formula is the unknown wingspan of the target, which is also to be estimated by the filters). These parameters are the inputs of the estimation.

III. Image processing

A. Getting images

To analyze and demonstrate the functionality of a vision based system it is important to have real or at least realistic camera images available. Three types of videos are used to test the developed image processing algorithm:

1) *Series of images* are generated by an appropriate simulator, which is able to provide realistic 3D views from the flight scenarios. Here we know the exact 3D positions and the size of the intruder and from these we can calculate the expected $d(t)$ and $\phi(t)$ at every t time instant.

2) *Shared videos from real UAV flights*, where the impact of the real environment can be observed but the position and size data of the intruder are unknown.

3) *Videos from real flight scenarios*, where the videos are recorded by real cameras and the position and size data of the intruder aircraft is known.

B. Images from flight simulator

The FlightGear flight simulator [6] was chosen to generate the realistic 3D environment for the indoor tests purpose, since it has a flexible interface with SIMULINK via the Aerospace blockset and the software is open source, hence the interface with the image processing algorithm can be customized as well.

The FlightGear software is adapted to get the rendered pictures into the main memory of the PC first. These pictures are then sent to MATLAB Engine. The image processing algorithm is done by this module. For the sake of calculating precise input data for the estimation algorithm the FlightGear program has to be calibrated. First the Field of View (*FoV*) and the aspect ratio settings are measured. For the measurements Cessna 172P aircraft model was used because this is a very popular light weight airplane. UAV share airspace with this type of aircrafts and most of them have no radar and use visual sensing for collision avoidance.

The wingspan of Cessna 172P is 11m. The *FoV* of the rendered image from the following model is calculated:

$$Fov = \frac{2\tan^{-1}(5.5/d)}{w_a} \cdot w \quad (3)$$

where *FoV* is in degree, *d* is the distance of the two aircrafts in meters, w_a is the measured width of the aircraft in pixels, *w* is the width of the rendered image in pixels.

From the measurements it turned out, that two regions can be defined from rendering point of view: a far region (d >20m), where this model can be used and a close region (d <20m), where distortions of this model are observed. The images can be used without additional compensation, since the far region is of interest in our case, because we are not dealing with the emergency situation yet. We have to detect the other aircraft in far enough to do the avoiding maneuver.

It also turned out that the FlightGear does not take care about the aspect ratio parameter. If geometry is not 1:1, the *FoV* is set to the bigger size and the image is cropped by FlightGear. According to the measurements that are not detailed here, it can be asserted that the geometry used by FlightGear is linear perspective.

C. Image processing algorithm

In this paper an image processing algorithm is presented which was designed to operate in daylight with clear or cloudy sky, when the contrast of the clouds is small or medium. When the contrast of the cloud is high (sunrise, sunset or storms), the vision algorithm cannot detect the intruder airplane robustly, however these situations can be predicted very well in advance. In our experimental environment the camera is fixed to the NED co-ordinate system.

From the very beginning of the algorithm design, we kept in mind the strict power, volume and other c of an air-born UAV application. To be able to fulfill these constraints, we decided to use many-core cellular array processor, implemented in ASIC or FPGA. Therefore we selected topographic operators, which well fit in this environment.

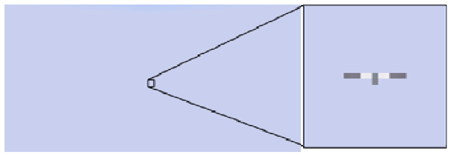


Fig. 3. Input image (2200x1100 pixel) from simulator; the square shows the location of the intruder, on the right side the enlarged image of the intruder

On Fig. 4 the flowchart of the image processing algorithm is shown. The input images of the algorithm are at least 1 megapixel (Fig. 3).

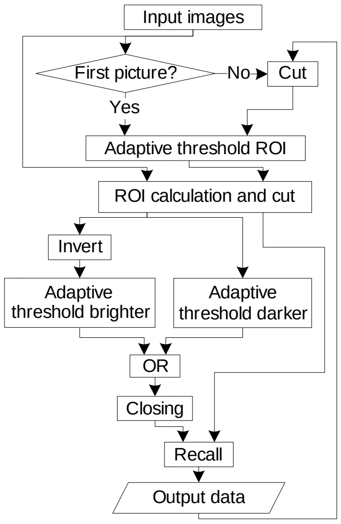


Fig. 4. Diagram of the image processing algorithm

As shown in Fig. 4 the first step is a space variant adaptive thresholding [10] to filter out the slow transitions in an image (Fig. 5). This can be the applied to the entire raw image if the position of the intruder is not known. If the location is already known, we track the intruder in a smaller window to reduce the data size and speed up the computation. The adaptive threshold results a binary image containing some of the points of the aircraft.

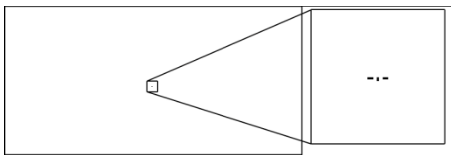


Fig. 5. Result of the first adaptive threshold on raw 2200x1100 input image; on the right side the enlarged image of the intruder aircraft

On this binary image a centroid calculation [10] is applied, which leads to the co-ordinates of the central pixel of the object. This co-ordinate will be the central pixel of the Region of Interest (ROI). The size of the ROI is determined by the previously calculated wing size plus 20 pixels in each direction. In that way two images are cut: one from the original picture (colored ROI image: Fig. 6a) and one from the result of the adaptive threshold (binary ROI image: Fig. 6b). The aircraft is composed of darker and brighter pixels than the intensity mean value of the original picture (background) (Fig. 3). On the colored ROI image two adaptive thresholding operators are calculated. The first one is calculated on the inverse picture of the grayscale image created from colored ROI image. With this threshold the pixels brighter than the intensity mean value of the original picture are found (Fig. 6c). The result is a binary image with the brighter pixels.

The other thresholding is calculated on the colored ROI image and with this the darker pixels are extracted (Fig. 6d). A logic OR is applied for the two thresholded images. The result is a binary picture with the found pixels of the aircraft and with some other pixels (Fig. 6e).

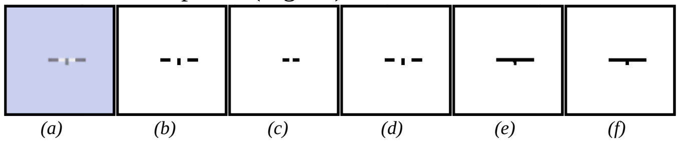


Fig. 6. The steps of the segmentation; from left to the right: a) color ROI, b) binary ROI, c) brighter pixels, d) darker pixels, e) OR operation and closing, f) segmented shape of the intruder aircraft

In some cases the parts of the airplane are not connected. A closing operation [10] is applied to connect the components. From the binary ROI picture we have an approximation for the aircraft and from the previously calculated picture we have the pixels of the whole airplane with some noise. As a last step, a recall operator [11] is applied, because the two adaptive thresholded (darker and brighter) may find other objects from the clouds, which are not extracted when the first adaptive threshold is applied.

The silhouette of the airplane is obtained in this way. On this picture the centroid in pixels is determined. Based on the co-ordinate of the center of the silhouette direction and the subtended angle of the intruder aircraft in radians can be determined accurately.

In the previous example, the intruder aircraft was in 1 km distance (60° view angle, 1200 pixels horizontal resolution, 1,02m/pixel), hence the extracted silhouette was very coarse. Here we show another example, where the intruder aircraft is only 300m form us (Fig. 7). It is observable in this snapshot that the first adaptive threshold does not find all of the pixels of the intruder (Fig. 7c) and the whole algorithm is needed to extract the entire aircraft.

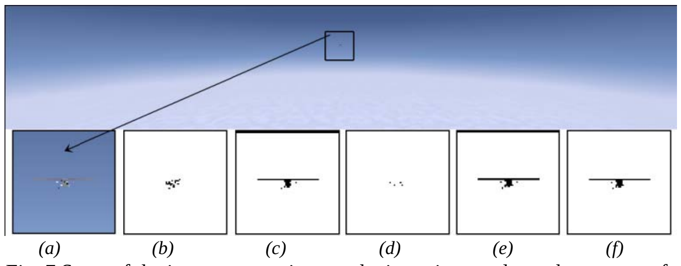


Fig. 7 Steps of the image processing: up the input image, down the outputs of each step: a) color ROI, b) adaptive threshold, c) darker pixels, d) brighter pixels, e) OR operation and closing, f) segmented aircraft

D. *Detection performance*

In our experimental settings, the intruder can be detected from 3.3km. In Fig. 8 the farthest detectable intruder is shown. In this case the size of the intruder aircraft is 2 pixels only.

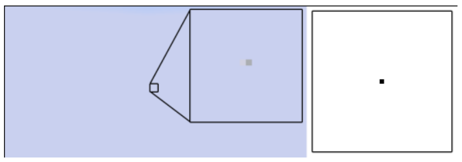


Fig. 8. Farthest detectable position of the intruder C172p aircraft (wingspan=11m), the distance is 3.3 km; on the left is the input image from FlightGear flight simulator, on the right the result of the segmentation

In Fig. 9 an example is shown with real image with cloudy background, when the contrast of clouds is middle. In Fig. 9 on the upper right the result of the first adaptive threshold is shown, from which the position of the intruder aircraft can be calculated.

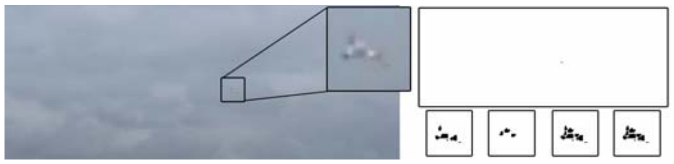


Fig. 9. Example of the situation with middle contrast clouds: on the left the original image with the enlarged aircraft; on the upper right the result of first adaptive threshold, on bottom right from left to the right the darker pixels, brighter pixels, OR operation and the segmented aircraft

In Fig. 10 we can see a typical situation during sunset, when the contrast of the clouds is high. In this case the position of the intruder can be determined only if we have prior information about it. In Fig. 10 on the upper right not only the points belonging to the intruder aircraft are detected by the first adaptive threshold but some cloud points also. On the bottom right the situation is shown when there is prior information about the position. This prior information may come from tracking or from a dispatcher. On the other hand, high contrast cloudy situations are known in advance (hence can be avoided), because it happens during sunrise, sunset, and in case of an approaching storm.

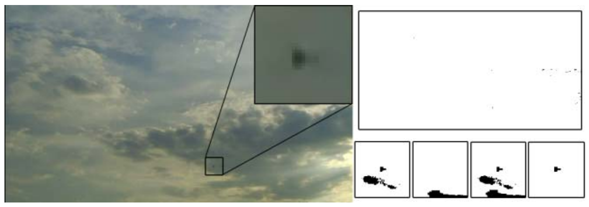
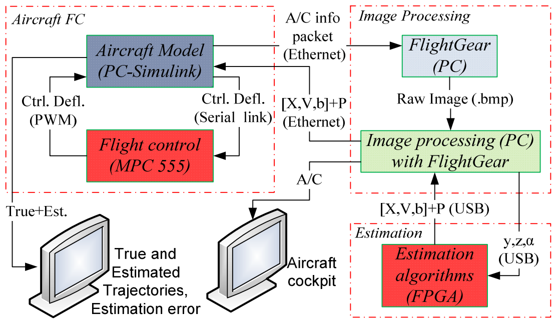


Fig. 10. Example of the situation with high contrast clouds: on the left the original image with the enlarged aircraft, on the upper right the result of first adaptive threshold, on bottom right from left to the right the darker pixels, brighter pixels, OR operation and the segmented aircraft

IV. THE EXPERIMENTAL SETUP

In Fig. 11 the diagram of the simulation environment is shown. The flight control is running on hardware in the loop system, shown at the upper left corner. The aircrafts are simulated by MATLAB/SIMULINK. For the own aircraft a high fidelity mathematical model has been identified using the measurement data collected from the Ultrastick unmanned aircraft [12]. The intruder is modeled as a simple double integrator. For the own aircraft a trajectory tracking controller has been designed, which runs on an MPC5200 embedded microprocessor. The flight simulator PC communicates with the image processing computer via Ethernet. On the image processing PC a modified FlightGear is running, which sends the rendered pictures to the MATLAB Engine and the results to an FPGA via USB. The FPGA realizes a Kalman filter and calculates the Motion Prediction data required by the control block. These data are forwarded to the control block by the image processing PC via Ethernet. Our aim is to implement the image processing algorithm on the FPGA in a later stage of the project, to reduce the power consumption together with the mass and volume of the system and to prepare it for onboard implementation of a UAV.


Aircraft FC
A/C info packet (Ethernet)
Image Processing
Aircraft Model (PC-Simulink)
FlightGear (PC)
Ctrl. Defl. (PWM)
Ctrl. Defl. (Serial link)
[X,V,b]+P (Ethernet)
Raw Image (.bmp)
Flight control (MPC 555)
Image processing (PC) with FlightGear
A/C
True+Est.
[X,V,b]+P (USB)
Estimation
y,z,α (USB)
Estimation algorithms (FPGA)
True and Estimated Trajectories, Estimation error
Aircraft cockpit
Fig. 11. Diagram of the Demo setup

V. CONCLUSIONS

In this article an image processing algorithm was presented for detecting intruder aircrafts in daylight situations with clear or cloudy sky when the contrast of clouds are small or medium. With this algorithm in the described environment, the position of the intruder aircraft can be determined with 1.02 m accuracy from 1km, given 60° view angle and 1024 pixels per line. In case of high contrast clouds more information is required for the accurate detection.

ACKNOWLEDGMENT

The ONR Grant (Number: N62909-10-1-7081) is greatly acknowledged.

REFERENCES

[1] Dempsey, M., “U.S. Army Unmanned Aircraft Systems Roadmap 2010-2035,” Tech. rep., U.S. Army UAS Center of Excellence, 2010.
[2] DeGarmo, M. T., “Issues Concerning Integration of Unmanned Aerial Vehicles in Civil Airspace,” Tech. rep., MITRE Center for Advanced Aviation System Development, 2004.
[3] Cox, T. H., Nagy, C. J., Skoog, M. A., Somers, I. A., and Warner, R., “Civil UAV Capability Assessment,” Tech. rep., NASA Dryden Flight Research Center, 2004.
[4] Hutchings, T., Jeffryes, S., and Farmer, S. J., “Architecting UAV sense & avoid systems,” *Proc. Institution of Engineering and Technology Conf. Autonomous Systems*, 2007, pp. 1–8.
[5] Fasano, G., Accardo, D., Forlenza, L., Moccia, A., and Rispoli, A., “A multi-sensor obstacle detection and tracking system for autonomous UAV sense and avoid,” XX *Congresso Nazionale AIDAA, Milano*, 2009.
[6] http://www.flightgear.org/Docs/getstart/getstart.html
[7] Stengel, R., *Flight Dynamics*, Princeton Press, 2004.
[8] B. Vanek, T. Péni, T. Zsedrovits, Á. Zarándy, J. Bokor and T. Roska., "Vision only Sense and Avoid system for small UAVs (Submitted for publication)", *Am.Control Conference 2011*, submitted for publication
[9] Hernandez, M. L., “Optimal Sensor Trajectories in Bearings-Only Tracking,” Tech. rep., QinetiQ, 2004.
[10] Pratt, W. K., *Digital Image Processing: PIKS Inside*, PixelSoft Inc., Los Altos, CA, 2001.
[11] T. Roska, L. Kék, L. Nemes and Á. Zarándy, *CNN Software Library, (Templates and Algorithms)*, Vol. DNS-1-1997, Comp. and Auto. Ins. of the Hung. Acad. of Sci., Budapest, Hungary, 1997.
[12] Paw, Y. C., *Synthesis and Validation of Flight Control for UAV*, Ph.D. thesis, University of Minnesota, 2009.