Moving Object Tracking using Simulated Motion Data

Şeyma BAYINDIR[1,2], Coşku KASNAKOĞLU[1]

[1] Department of Electrical and Electronics Engineering,
TOBB University of Economics and Technology, Ankara, Turkey.
[2] TUBİTAK SAGE, Ankara Turkey.
seymabayindir1@gmail.com, kasnakoglu@etu.edu.tr

Abstract

This paper presents the development of missile motion tracking using computer simulation and analyzes the results. In order to investigate the missile motion, simulated data and missile model is implemented in numerical computing package MATLAB/Simulink. To analyze the motion, object detection, corrections of the calculations using Kalman filter and velocity estimation of detected object steps are implemented on simulated data. The generated results are compared and shown at the end of the paper.

1. Introduction

Image processing techniques are used by many applications for object detection, tracking and position estimation [1][2]. The object of interest can be static or dynamic depending on operation. Moving object detection is more challenging than the static object detection [3]. In this task, the object is dynamic and tracked as long as the object is visible on the camera.

Due to growing camera capabilities, image processing results have become useful in different purposes. Better camera resolution also provides more precise results. Cameras are now commonly assembled in air vehicles like airplanes and missiles after camera cases became more resistant to pressure and temperature [4]. Instead of employing disposable sensors on the jettisoned missile, the camera data can be used to build the necessary results, thus the cost is reduced.

The missiles have to be tested many times after design process [5]. To validate that the movement of missile is as expected, the camera can be placed at the suitable position. The aircraft is the only choice for the position of camera. In the real life, the field of view and placement of cameras under the plane is a very important factor for the video quality [6]. The decision of the camera placement and field of view is quite difficult because of the changing environmental conditions like the direction of the sun light [7].

In this research, simulated video data is utilized to help camera placement while avoiding such real environment disadvantages. Then an object detection algorithm has been implemented based on video data. After extracting the missile as the object of interest, the trajectory of the missile is generated. To provide more accurate results a Kalman filter is implemented [8]. Velocity estimations are generated for the evaluating the results. The real life motivation behind the study is to be able to track a missile dropped from a fighter jet, similar to the scenario shown in Fig. 1.

2. Implementation of Missile Model and Development of Simulated Motion Data

The simulated environment, missile model and plane model have been built in MATLAB/Simulink/V-Realm Building Tool as shown in Fig. 2. The camera has been placed at different positions under the plane. Then camera direction has been changed to find the best field of view and position of camera. The captured image from cameras placed at different locations are given in Fig. 3, Fig. 4, Fig. 5 and Fig. 6.

Fig. 1. A Fighter Jet Dropping a Missile

To decide on the camera position, two factors have been considered. These factors are:

1. The length of video time range in which the missile can be seen.
2. The visibility quality of the missile in the video.

Based on these criteria the last camera position is selected and the videos are generated from that camera position for the rest of this paper. The frames captured from this camera at different times are given in Fig. 7 and Fig. 8. After creating the environment and the models, the simulated video data has been created in the MATLAB/Simulink environment. The plane and environment models are created as described in [9].

To simulate the motion, the plane's x , y , z position parameters are imported from a F-16 jet fighter translational data. The raw data has latitude, longitude and barometric altitude values. Thus first the latitude and longitude values are converted from degrees to Universal Transverse Mercator (UTM) format. Then the converted data initial positions are set to normalized to (0,0,1).

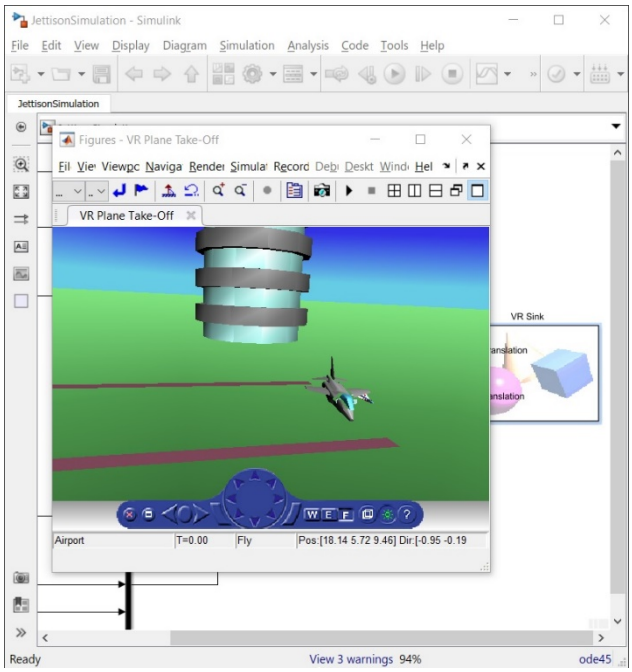

Fig. 2. MATLAB/Simulink Environment for Simulated Video Data Generation

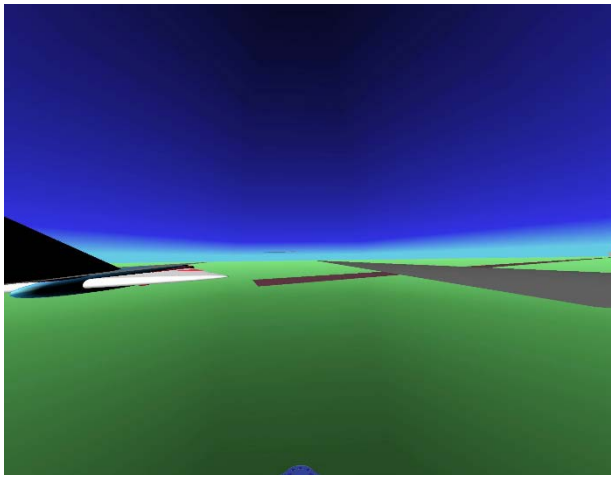

Fig. 3. The Image Captured From Camera Position-1

The missile motion is implemented in two parts. At the first part of the motion, the missile moves within the plane. So, the translation of the missile is the same as the plane translation. At the second part, the missile has been jettisoned and the altitude parameter decreases.

In the Simulink environment, a model which receives all parameters from a file and gives a video as an output has been implemented. The video length is 13 seconds. The frame size is 1480 x 1971.

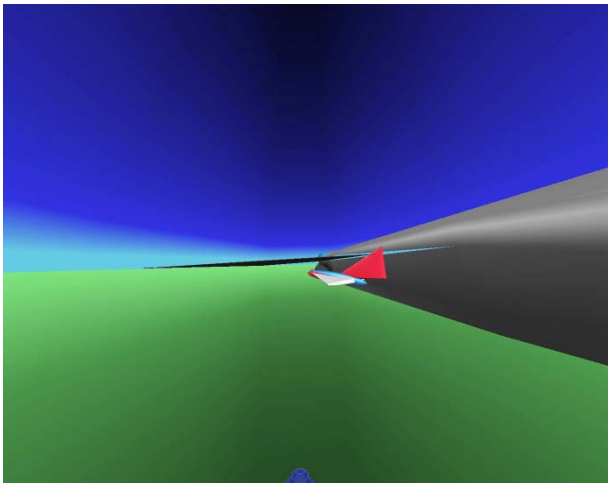

Fig. 4. The Image Captured From Camera Position-2

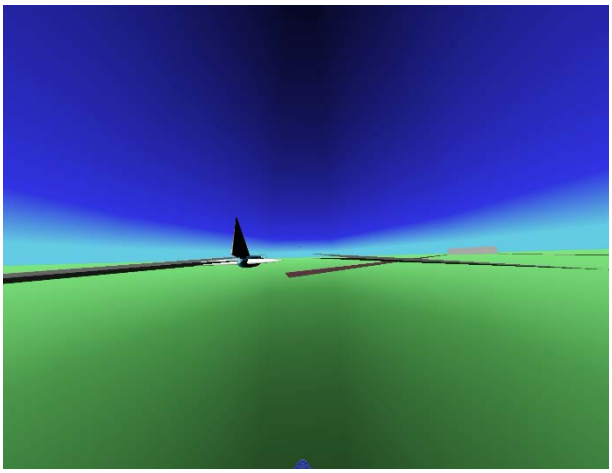

Fig. 5. The Image Captured From Camera Position-3

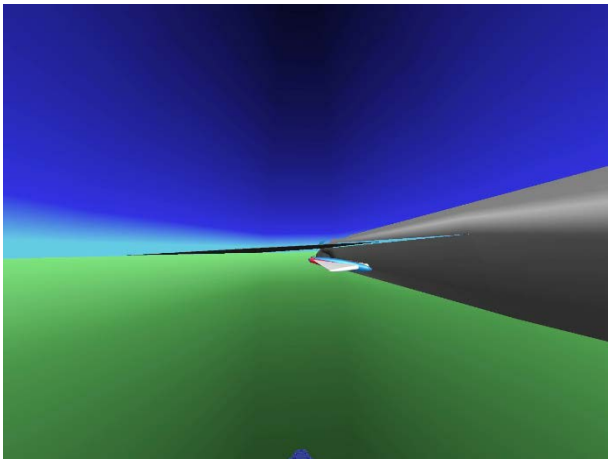

Fig. 6. The Image Captured From Camera Position-4

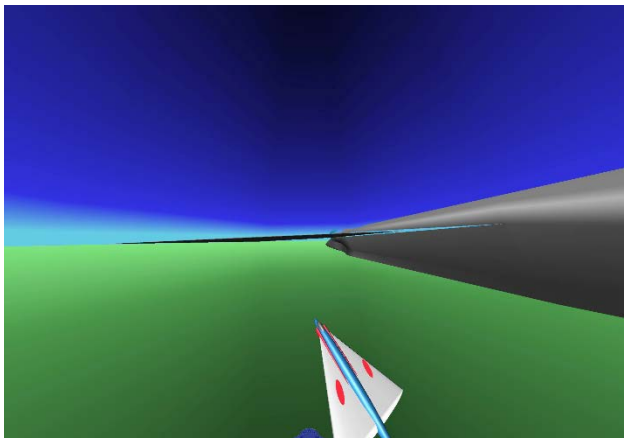

Fig. 7. The Image Captured From Camera Positioned at Selected Coordinates

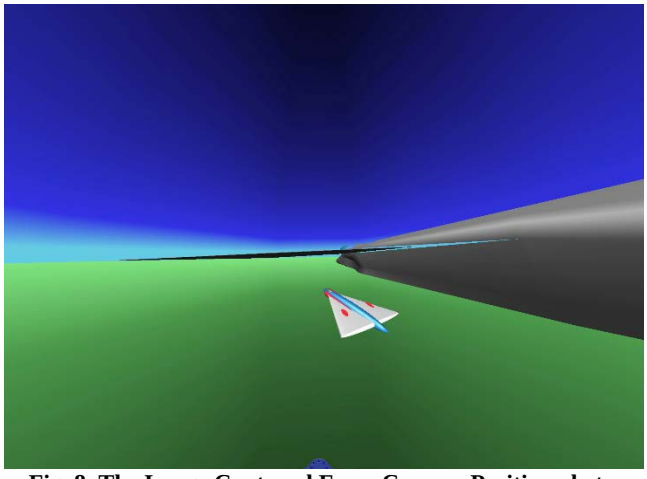

Fig. 8. The Image Captured From Camera Positioned at Selected Coordinates

3. Object Detection

Object detection algorithms have mainly two purposes: Background extraction and object detection. The number of training frames are selected according to the target motion in the video. After extracting the foreground of video, a blob which determines the connected pixels is detected.

In this study, the missile is selected as the object and an object detection method is implemented for it. The foreground is modeled as a Gaussian mixture. The starting 150 frames are used as the training frames. After extracting the foreground, the missile velocity is assumed constant and the centroid of the missile is estimated at each frame. The minimum number of pixels is 100. So, the blobs which have less than 100 connected pixels are not evaluated.

4. Kalman Filter Implementation to Simulated Video

Kalman filter is commonly used in many applications including control, navigation and computer vision. Kalman filtering method is a recursive estimation method. It is used to decrease the prediction error is the presence of model uncertainties and noises.

Let us start from the following state space model of the system

$$\begin{cases} \underline{x}(k+1) = \Phi(k+1,k)\underline{x}(k) + \Gamma(k+1,k)\underline{w}(k) \\ \qquad\qquad + \Psi(k+1,k)\underline{u}(k) \\ \underline{z}(k+1) = H(k+1)\underline{x}(k+1) + \underline{v}(k+1) \end{cases}$$

where $\underline{x}(0)$ is the initial state of the system with the following first and second order statistics:

$$E\left[\underline{x}(0)\right] = \underline{m}_x(0)$$

$$\operatorname{cov}\left[\underline{x}(0)\right] = E\left[\left(\underline{x}(0) - \underline{m}_x(0)\right)\left(\underline{x}(0) - \underline{m}_x(0)\right)^T\right] = P_x(0)$$

Here $\{\underline{w}(k)\}$ is the process white Gaussian noise with zero mean and covariance $Q(k) \geq 0$ and $\{\underline{v}(k)\}$ is the measurement white Gaussian noise with zero mean and covariance $R(k) \succ 0$. It is assumed that $\{\underline{w}(k), \underline{v}(k), \underline{x}(0)\}$ are uncorrelated for any k.

The state estimation process is a cycle carried out in two steps:

1) Time update (prediction):

$$\begin{cases} \hat{\underline{x}}(k+1|k) = \Phi(k+1,k)\hat{\underline{x}}(k|k) \\ \qquad\qquad + \Psi(k+1,k)\underline{u}(k) \\ P(k+1|k) = \Phi(k+1,k)P(k|k)\Phi^T(k+1,k) \\ \qquad\qquad + \Gamma(k+1,k)Q(k)\Gamma^T(k+1,k) \end{cases}$$

2) Measurement update (correction):

$$\begin{cases} \hat{\underline{x}}(k+1|k+1) = \hat{\underline{x}}(k+1|k) \\ \qquad\qquad + K(k+1)\left[\underline{z}(k+1) - H(k+1)\hat{\underline{x}}(k+1|k)\right] \\ K(k+1) = P(k+1|k)H^T(k+1) \times \\ \qquad\qquad \left[H(k+1)P(k+1|k)H^T(k+1) + R(k+1)\right]^{-1} \\ P(k+1|k+1) = \left[I - K(k+1)H(k+1)\right]P(k+1|k) \end{cases}$$

The initialization is performed as follows:

$$\hat{\underline{x}}(0|0) = \underline{m}_x(0)\ ;\ P(0|0) = P_x(0)$$

The following special case is also useful

$$\begin{cases} \underline{x}(n) = \Phi\underline{x}(n-1) + \underline{w}(n) \\ \underline{y}(n) = H\underline{x}(n) + \underline{v}(n) \end{cases}$$

$$E\left[\underline{x}(0)\right] = \underline{m}_X(0)$$

$$\operatorname{cov}\left[\underline{x}(0)\right] = E\left[\left(\underline{x}(0) - \underline{m}_X(0)\right)\left(\underline{x}(0) - \underline{m}_X(0)\right)^T\right] = P_X(0)$$

where $\{\underline{w}(n)\}$ is white noise with zero mean and

$$E\left[\underline{w}(n)\cdot\underline{w}^T(n)\right]=Q\ ;\ E\left[\underline{w}(n)\cdot\underline{w}^T(m)\right]=0\ \forall m\neq n$$

$\{\underline{v}(n)\}$ is white noise with zero mean and

$$E\left[\underline{v}(n)\cdot\underline{v}^T(n)\right]=R\ ;\ E\left[\underline{v}(n)\cdot\underline{v}^T(m)\right]=0\ \forall m\neq n$$

The noises and the initial state are uncorrelated so

$$E\left[\underline{v}(n)\cdot\underline{w}^T(n)\right]=E\left[\underline{x}(0)\cdot\underline{v}^T(m)\right]$$
$$=E\left[\underline{x}(0)\cdot\underline{w}^T(m)\right]=0\ \forall m,n$$

It can be shown that the optimal solution is then:

1) Initialization:

$$\hat{\underline{x}}(0\,|\,0)=\underline{m}_x(0)\ ;\ P(0\,|\,0)=P_x(0)$$

2) Time update (prediction):

$$\begin{cases}\hat{\underline{x}}(n\,|\,n-1)=\Phi\hat{\underline{x}}(n-1\,|\,n-1)\\ P(n\,|\,n-1)=\Phi P(n-1\,|\,n-1)\Phi^T+Q\end{cases}$$

3) Measurement update:

$$\begin{cases}\hat{\underline{x}}(n\,|\,n)=\hat{\underline{x}}(n\,|\,n-1)\\ \qquad +K(n)\left[\underline{y}(n)-H\hat{\underline{x}}(n\,|\,n-1)\right]\\ K(n)=P(n\,|\,n-1)H^T\left[HP(n\,|\,n-1)H^T+R\right]^{-1}\\ P(n\,|\,n)=\left[I-K(n)H\right]P(n\,|\,n-1)\end{cases}$$

In this study, the Kalman filtering process summarized above is implemented for predicting the position of the missile based on simulated video data. Fig. 9 and Fig. 10 shows the estimated object detection algorithm and Kalman filtering results. According to that result, Kalman filter decreases the error. The missile moves with constant velocity and the sudden movements are not expected. The object detection results have large deviations at some time. Kalman filter corrects these deviation.

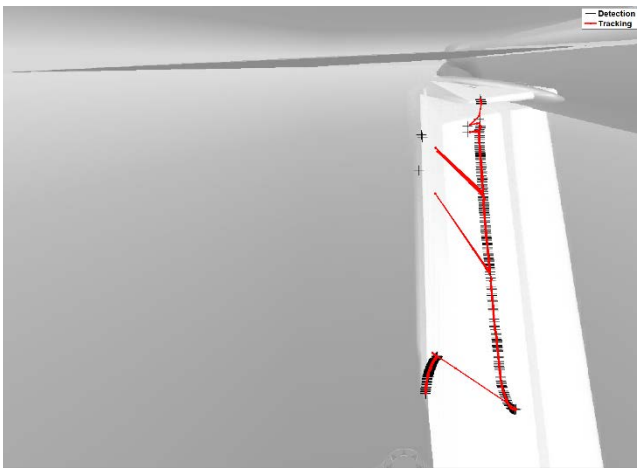

Fig. 9. The Motion Graph of Missile

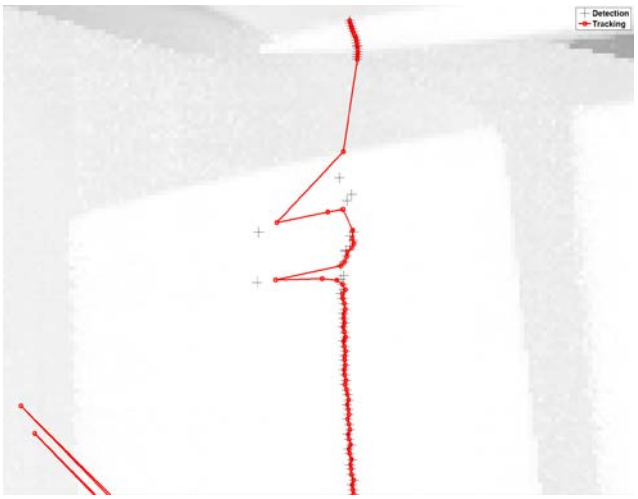

Fig. 10. The Zoomed Motion Graph of Missile

5. Calculate the Missile Velocity

The velocity estimation is used for validating the advantages of the Kalman filter usage in this task. After detection of missile and determination of the missile center position in the frame, Euclidean distance formula has been used to estimate the object position difference from two frames

$$\Delta D\ =\ \sqrt{(X_2-X_1)^2\ +\ (Y_2-Y_1)^2}$$

where X_1 is the previous horizontal position of the object center, X_2 is the initial horizontal position of the object center, Y_1 is the previous vertical position of the object center, Y_2 is the initial vertical position of the object center. The velocity is then calculated as the ratio of distance to time.

$$V\approx\frac{\Delta D}{\Delta \mathrm{t}}.$$

Two velocity graphs of the missile are given in Fig. 11 and Fig. 12. The former shows the missile velocity based on raw data before implementing any algorithm. The data resolution is lower than that of the video resolution. Because of the resolution difference, the velocity is computed as zero at some time intervals. To prevent such spikes in velocity, the Kalman filter smooths the velocity results. The velocity graph after implementing the Kalman filter is shown in Fig. 12. Clearly erroneous zero velocity values have been eliminated.

6. Conclusions and Future Works

A tracking algorithm and Kalman filter are implemented based on simulated video data. The trajectory and velocity calculation results are the outputs of this study. According to the results, Kalman filter gives better and smoother estimates. Generally, high speed cameras are used for airborne applications. Because of the high frame rate of high speed cameras, sudden movements are not expected. However, a standard object detection approach can wrongfully detect such instantaneous object position changes. Kalman filtering provides a smooth trajectory and predicts the object position in such ambiguous situations.

Future directions include integrating the method developed in this paper to other works carried out by our research group at

TOBB ETU, such as corner detection, flight stabilizing, hardware-in-the-loop testing and emergency flight recovery [10]-[15].

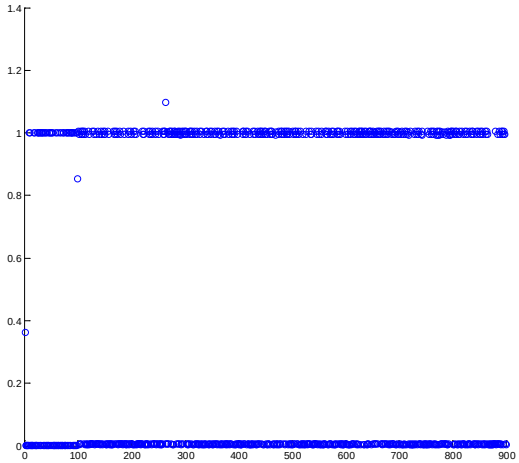

Fig. 11. The Velocity Graph Calculated from Raw Data

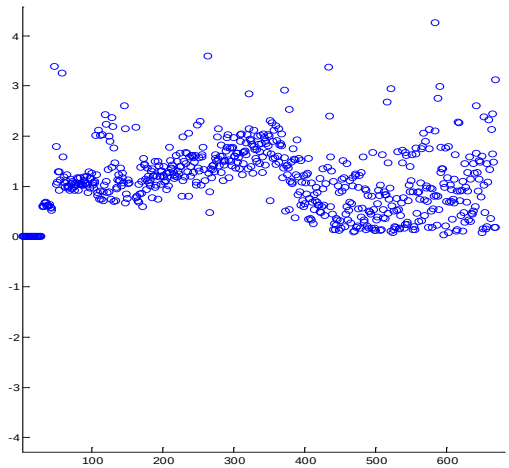

Fig. 12. The Velocity Graph Calculated from Processed Data

7. References

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. "You only look once: Unified, real-time object detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[2] Sivasankaran, S., Mohandas, S., & Paulose, M. "Tracking Location of Medicinal Plants in Dense Forest Using Image Processing: A Comparative Analysis." Journal of Emerging Technologies and Innovative Research. Vol. 3. No. 10 (October-2016). JETIR, 2016.

[3] Chavez-Garcia, R. O., & Aycard, O. "Multiple sensor fusion and classification for moving object detection and tracking." IEEE Transactions on Intelligent Transportation Systems 17.2 (2016): 525-534.

[4] Popov, A., Miller, A., Miller, B., Stepanyan, K., Konovalenko, I., Sidorchuk, D., & Koptelov, I. "UAV navigation on the basis of video sequences registered by onboard camera." Proceedings of the 40th IITP RAS Interdisciplinary Conference and School, St. Petersburg, Russia. 2016.

[5] Chen, W., Zeng, X. G., Ren, Y. Q., & Zhao, S. Y. "Research on Evaluation Index System of Air to Ground Missile Testing Quality." MATEC Web of Conferences. Vol. 63. EDP Sciences, 2016.

[6] Andert, F., Ammann, N., Puschel, J., & Dittrich, J. "On the safe navigation problem for unmanned aircraft: Visual odometry and alignment optimizations for UAV positioning." Unmanned Aircraft Systems (ICUAS), 2014 International Conference on. IEEE, 2014.

[7] Chen, X., & Guo, J. "Missile loader manipulator positioning technology based on visual guidance." Control, Automation and Robotics (ICCAR), 2016 2nd International Conference on. IEEE, 2016.

[8] Brookner, Eli. Tracking and Kalman filtering made easy. John Wiley & Sons, Inc., 1998.

[9] The Mathworks, Plane Take Off Example, https://www.mathworks.com/help/sl3d/examples/plane-take-off.html, 2017.

[10] Korkmaz, H., Ertin, O. B., & Kasnakoğlu, C. Korkmaz. "Design of a flight stabilizer system for a small fixed wing unmanned aerial vehicle using system identification." IFAC Proceedings Volumes 46.25 (2013): 145-149.

[11] Aydogdu, M. F., Demirci, M. F., & Kasnakoglu, C., Aydogdu, M. Fatih, M. Fatih Demirci, and Cosku Kasnakoglu. "Pipelining Harris corner detection with a tiny FPGA for a mobile robot." Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on. IEEE, 2013.

[12] Atlas, E., Erdoğan, M. I., Ertin, O. B., Güçlü, A., Saygi, Y. E., Kaynak, Ü., & Kasnakoğlu, C. "Hardware-in-the-loop test platform design for UAV applications." Applied Mechanics and Materials. Vol. 789. Trans Tech Publications, 2015.

[13] Ertin, O. B., Korkmaz, H., Kaynak, U., & Kasnakoglu, C. "Hardware-in-the-loop test platform for a small fixed wing unmanned aerial vehicle embedded controller." The 32nd IASTED International Conference on Modelling, Identification and Control (MIC 2013), Innsbruck, Austria. 2013.

[14] Akyurek, S., Ozden, G. S., Kurkcu, B., Kaynak, U., & Kasnakoglu, C. Akyurek, Seyma, et al. "Design of a flight stabilizer for fixed-wing aircrafts using H∞ loop shaping method." Electrical and Electronics Engineering (ELECO), 2015 9th International Conference on. IEEE, 2015.

[15] Kasnakoglu, C., & Kaynak, U. "Automatic recovery and autonomous navigation of disabled aircraft after control surface actuator jam." AIAA Guidance, Navigation and Control Conference, Toronto, Canada. 2010.