UNLV | UNIVERSITY LIBRARIES

Electrical and Computer Engineering Faculty Publications

Electrical & Computer Engineering

2010

Real-time On-board Object Tracking for Cooperative Flight Control

Ajay Kumar Mandava
University of Nevada, Las Vegas, mandavaa@unlv.nevada.edu

Emma Regentova
University of Nevada, Las Vegas, emma.regentova@unlv.edu

Henry Selvaraj
University of Nevada, Las Vegas, henry.selvaraj@unlv.edu

Follow this and additional works at: https://digitalscholarship.unlv.edu/ece_fac_articles

Part of the Controls and Control Theory Commons, Multi-Vehicle Systems and Air Traffic Control Commons, Signal Processing Commons, and the Systems and Communications Commons

Repository Citation

Mandava, A. K., Regentova, E., Selvaraj, H. (2010). Real-time On-board Object Tracking for Cooperative Flight Control. *Systems Science, 36*(2), 15-22.
https://digitalscholarship.unlv.edu/ece_fac_articles/276

This Article is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Article in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Article has been accepted for inclusion in Electrical and Computer Engineering Faculty Publications by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

Systems Science, Vol. 36, No. 2, 2010

AJAY KUMAR MANDAVA*, EMMA E. REGENTOVA*,
HENRY SELVARAJ*

REAL-TIME ON-BOARD OBJECT TRACKING FOR COOPERATIVE FLIGHT CONTROL

One of possible situations for cooperative flights could be a scenario when the decision on a new path is taken by a certain fleet member, who is called the leader. The update on the new path is transmitted to the fleet members via communication that can be noisy. An optical sensor can be used as a back-up for re-estimating the path parameters based on visual information. For a certain topology, the problem can be solved by continuous tracking of the leader of the fleet in the video sequence and re-adjusting parameters of the flight, accordingly. To solve such a problem a real time system has been developed for recognizing and tracking 3D objects. Any change in the 3D position of the leading object is determined by the on-board system and adjustments of the speed, pitch, yaw and roll angles are made to sustain the topology. Given a 2D image acquired by an on-board camera, the system has to perform the background subtraction, recognize the object, track it and evaluate the relative rotation, scale and translation of the object. In this paper, a comparative study of different algorithms is carried out based on time and accuracy constraints. The solution for 3D pose estimation is provided based on the system of Zernike invariant moments. The candidate techniques solving the complete set of procedures have been implemented on Texas Instrument TMS320DM642 EVM board. It is shown that 14 frames per second can be processed; that supports the real time implementation of the tracking system with the reasonable accuracy.

1. Introduction

The system is designed for detecting a specific flying object and distinguishing changes in the 3D rotational angles under different scales and translations from 2D images only. Specifically, we consider a situation when other sensors malfunction or are jammed and/or the communication is lost or compromised. The problem also is related to a selection of a minimum set of sensors to be used as small flying objects such as mini- and medium-sized Unmanned Aerial Vehicles (UAV) have limited payload capability. Although 2D image data cannot deliver the wealth of knowledge about 3D pose, it allows for approximating the path based on continuous tracking and capturing the change when it accumulates to a certain threshold value. By doing this, a "slave" object will get the information about relative change of the path of the leading object, which can be an aircraft, a fighter or another UAV of larger scale. Based on the information a "slave" object in a fleet recalculates its own flight parameters and sustains a prescribed topology of the group.

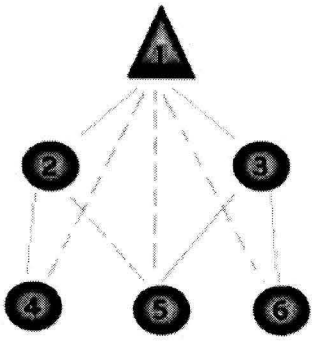

Fig. 1. Cooperative flight topology.

An example of a topology is shown in Fig. 1 that allows for establishing different leaders as shown by solid and dashed lines. For example, the leadership relations can be established as 2-1, 3-1, 4-2 (or 4-1), 6-3 (or 6-1), 5-3, (or 5-1, or 5-2). These relations are established by steering the camera into the fixed point in 3D, where the leading object is expected to be present according to the initial topology assignment. Each of the slave objects is to see the leader within this location and at the same pose within the frame image

* Department of Electrical and Computer Engineering, University of Nevada, Las Vegas, 4505 Maryland Parkway, Box 454026 Las Vegas, NV 89154-4026, e-mail: regent@ee.unlv.edu

acquired by the on-board camera. Figure 2 exhibits the idea. The figure shows the initial and changed poses of the leader and the pose as seen by the slave after its own pose/speed are corrected. Because the pose is estimated with respect to the camera coordinates, the actual parameters are to be recalculated with respect to the object coordinates.

Fig. 2. Left: expected pose of the leader; center: change of the pose; right: when the pose is corrected.

The complete procedure can be described as a sequence of steps as in Fig. 3. Though we expect the leading object to be in a predetermined position, there is a need to recognize it correctly. Only then is the object tracked continuously, and the pose is estimated in a continuous manner, facilitating more accurate calculation of the flight parameters. If the object cannot be tracked correctly, the detection is invoked again and the loop continues.

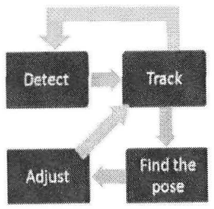

Fig. 3. System overview.

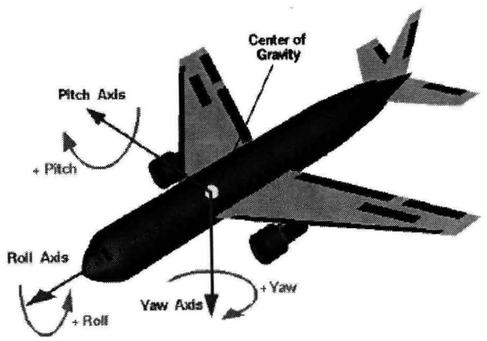

Fig. 4. Pitch, Roll and Yaw.

Detection and pose estimation of 3D objects is given considerable attention in literature. Some methods are directly derived and tested for aircraft detection. In the method described by Dudani [1, 21], aircraft pose is estimated using the Hu moments and the nearest neighbour search. The class recognition performance is approximately 95%. The training set consists of approximately 500 images for each of the six aircraft types. The problem of pose ambiguity is not dealt with; it is simply assumed that this can be solved.

Breitenstein and Kuettel [2] have developed a method based on a novel error function that compares the input range image to pre-computed pose images of an average face model. The method is shown to be robust for large pose variations of ±90° yaw, ±45°, pitch and ±30° roll rotation, facial expression, partial occlusion, and works for multiple faces in the field of view. It correctly estimates 97.8% of the poses within yaw and pitch error of 15° at 55.8 fps.

Osadchy in [3] has proposed a system that integrates detection and pose estimation of faces by training a Convolutional Neural Network (CNN) to map faces to points on a manifold, parameterized by pose, and non-faces to points far from the manifold and used 52,850 face images for training. It classifies 80% of the yaw rotations within 15° error at 5 fps.

Yuan and Niemann [4] developed an appearance based neural approach for this task. First, the object is represented in a feature vector derived by a principal component network. Then, the NN classifier trained with a resilient back propagation algorithm is applied to identify it. Next, pose parameters are obtained by four NN estimations. The NNs have been trained on the same feature vector. Under occlusion and noise, the average recognition rate is 77%.

Chang and Ghosh [5] proposed a scheme using spherical manifolds for simultaneous classification and pose estimation of 3-D objects from 2-D images. 10 different aircrafts and a total of 684 pose images at 128×128 pixel resolution have been used for pose estimation and it is capable of giving reliable pose estimates (within 40° accuracy) for almost 70% of the data, or within 20° for over 50% of the data, even when rotation invariant features were used.

In this paper, we show how the problem can be solved at the system level; and provide the hardware implementation for the real-time systems; the accuracy of pose estimation and the time to complete the task is estimated. The paper is organized as follows: Section 2 describes the procedures and methods for implementing the pipeline shown in Fig. 3. Experiments and test results are presented in Section 3. The hardware implementation of candidate methods is described in Section 4 and Section 5 concludes the paper.

2. Methodology

To implement the flowchart in Fig. 3, we have studied the performance of various techniques for background subtraction, object identification, tracking and pose estimation. The effectiveness of these techniques is discussed. The aim is to find a simple yet robust solution for a real-time implementation. The different methods studied and implemented for different modules are discussed below

Background Subtraction

Background subtraction is a critical step in video processing pipeline for applications such as surveillance, tracking or pose estimation. Difficulties in finding the object pose can arise due to wind, rain or illumination changes brought by weather, background variations and the path change. Finding the object in a different location indicates relative motion and prompts to switch to the pose estimation procedure.

The goal of the system is to reach a balance between robustness and computation cost. To achieve the goal, four different methods are studied and implemented for background subtraction. The algorithms are:

- Frame differencing [6, 7].
- Approximated median filter [8, 9, 23–26].
- Mixture of Gaussians (MoG) [10].

Frame differencing is arguably the simplest background modeling technique; frame differencing uses the video frame at time $t-1$ as the background model for the frame at time t.

Median filtering is one of the most commonly used background modeling techniques. The background estimate is defined to be the median at each pixel location for all the frames in the buffer.

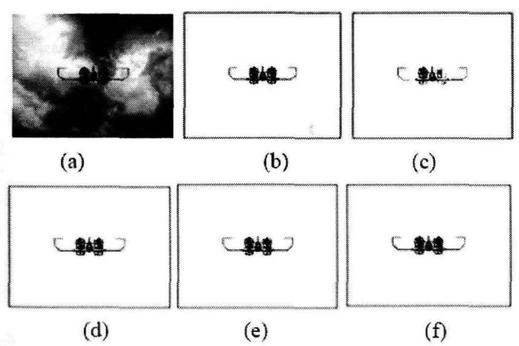

Fig. 5. (a) Original image, (b), (c), (d) frame subtraction (with various thresholds), (e) approximated median filter, (f) mixture of Gaussians.

MoG maintains a density function for each pixel. Thus, it is capable of handling multi-model background distributions. On the other hand, since MoG is parametric, the model parameters can be adaptively updated without keeping a large buffer of video frames.

Figure 5 shows the results of various background subtraction techniques.

The frame differencing may yield a very noisy result and the quality depends on the threshold parameter. The method based on an approximate median, produces better accuracy under a moderate computation cost. It encounters a problem with quickly changing light levels, but handles them better than mixture of Gaussians. The latter has a good performance, but presents a tricky parameter optimization problem. Therefore, the approximate median filter, which is also computationally moderate is seen as a good choice for hardware implementation.

Object Identification

Object recognition is a challenging task because of wide variability of objects that can be encountered in the scene, for example, in combat operations. For feature based methods, one of many challenges is the decision of features to be used and their computation.

For object detection, we have analyzed the performance of invariant moments (Hu [1], Geometric [22], Zernike [11]), scale invariant feature transform (SIFT) [12] and affine scale invariant feature transform (ASIFT) [13] methods. SIFT is fully invariant with respect to only four parameters, namely zoom, rotation in 2D and translation, the ASIFT method treats the two left over parameters: the angles defining the camera axis orientation and retrieves the object even under extreme angle. In technical terms, ASIFT is affine invariant of SIFT.

In the first step, the object image is acquired by using the camera mounted on the UAV following the leader UAV and the background is subtracted. Then, a set of moment invariants (SIFT or ASIFT features for the background subtracted images) is extracted after binarization of background subtracted image. In the identification phase, the object image is classified by pairwise comparison of extracted features and database values or features. For moments, the one with minimum Euclidean distance is the identified object and for SIFT and ASIFT, the one with maximum feature mapping is considered as identified object.

Tracking

Visual tracking with low computational complexity is a goal. It substitutes segmentation of each frame which is computationally expensive. Our aim is to find solutions that are robust, simple, computationally

feasible, modular, and easily adaptable to various applications.

Various tracking algorithms have been evaluated: mean shift with variable and fixed sized windows [14–17], the scale invariant feature transform, Harris [18] and fast full search based on fast Fourier transform algorithms [19]. Among all the methods, SIFT features are scale invariant and more robust to illumination changes compared to other algorithms and quite stable to occlusions. However, compared to mean shift and SSD FFT algorithms, SIFT is computationally intensive. Thus, for real time implementation we considered the mean shift algorithm based on time/accuracy constraints.

Pose estimation

In its simplest form, pose can be defined as the problem of estimating the three angles of an object (as shown in Fig. 4) in the image plane as it moves around a scene. In our application, object pose can change due to object motion, changing appearance patterns of the object and the scene, object-to-object and object-to-scene occlusions and camera motion.

In order to obtain the pose of the object for a given image, it should be compared with the stored images of different targets at various orientations. Storing the whole image of each view of the object and subsequent comparison with the given image is physically not possible due to memory and time limitations. Instead, we can store moment features of the target at different orientations. For this task, we tested three sets of moment functions, i.e., Hu [1], Complex Radial [22] and Zernike [11] for the accuracy.

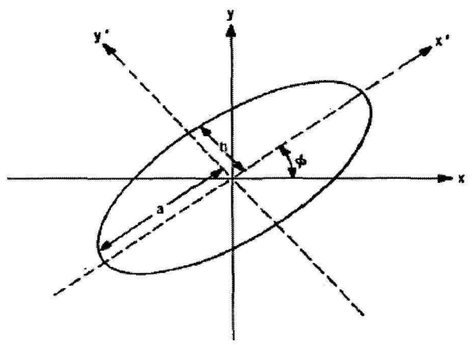

Fig. 6. Tilt angle calculation.

The pose estimation is performed based on the following procedures. We populate a database (a look up table) with the moments calculated for objects with a tilt angle (ϕ) and three other varying angles (pitch, roll, and yaw) per object (see Fig. 6). Based on the moment values calculated from an entry image, we calculate the tilt angle according to equation (1) and Table 1.

$$\phi = \frac{1}{2}\tan^{-1}\left(\frac{2\mu_{11}}{u_{20}-\mu_{02}}\right). \quad (1)$$

The moment multiplied by the tilt angle is used as an index to the look-up table. Then, we calculate distances between moments per corresponding entry to select an entry of a minimum Euclidean distance. The scale is calculated as

$$\text{scale} = \frac{m'_{00}}{m_{00}},$$

where m'_{00} and m_{00} are the central moments for objects in the look-up table and the test image, respectively.

Table 1. Angle calculation from moment functions.

$u_{20}-\mu_{02}$	μ_{11}	ϕ	
0	0	0	$\varepsilon = \frac{2\mu_{11}}{u_{20}-\mu_{02}}$
0	+	+45°	
0	–	–45°	
+	0	0	
–	0	–90°	
+	+	$\frac{1}{2}\tan^{-1}(\varepsilon)$	$0 < \phi < 45°$
+	–	$\frac{1}{2}\tan^{-1}(\varepsilon)$	$-45° < \phi < 0$
–	+	$\frac{1}{2}\tan^{-1}(\varepsilon)+90°$	$45° < \phi < 90°$
–	–	$\frac{1}{2}\tan^{-1}(\varepsilon)-90°$	$-90° < \phi < -45°$

3. Experimental tests and results

For experiments we populate three databases. All the images are of 600×800 pixels. Database-1 contains 1331 images per object from the objects in Fig. 7 of different Pitch, Yaw, and Roll angles in combination from 0~40°, with a minimal step of 4 degrees. Thus, there are six possible values: 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40 for Pitch, Yaw, and Roll, respectively, hence a total of 1331 training images.

Database-2 contains 343 images per object from those in Fig. 2 of different Pitch, Yaw, and Roll angles in combination from 0~40°, with a minimal step size of 6 for each angle. Thus, there are six possible values: 0, 6, 12, 18, 24, 30, 36 for Pitch, Yaw, and Roll, hence a total of 343 training images.

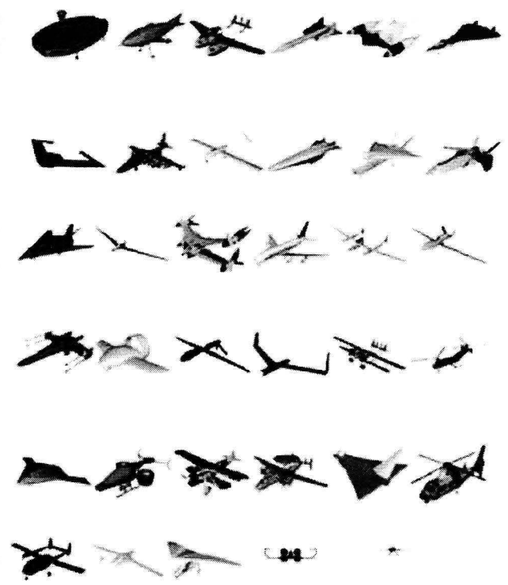

Fig. 7. Test objects.

Database-3 contains 35 different object images of 5 different views, hence a total of 175 images. Database-1 and Database-2 are used for pose estimation and Database-3 for object identification. Each lookup table entry corresponds to one of the images in the database. For each lookup table entry, the index value of the object, the values of absolute moment invariants, the 2D rotational angle θ from the object's principle axes, and the scaling factor of the corresponding database image are tabulated.

Each entry is considered as an object of a specific shape whose descriptors are compared based on the Euclidean distance. A minimum distance comparison yields a specific pose.

The following tests are performed:

Test 1

The scale of the original test object is varied from 0.1 to 1.5. We have randomly chosen 300 images from the database for object detection and pose identification.

Test 2

To study the effect of noise, we choose several types of noise:

- A Gaussian additive noise of zero mean and variance of 0.01 to 0.20 with an increment of 0.01. A total of 300 test images have been chosen for the object detection and identification of pose.
- A speckle multiplicative noise is introduced to image I, as $J = I + n*I$, where n is uniformly distributed random noise with mean 0 and variance V. V varies from 0.01 to 0.20 with an increment of 0.01. A total of 300 test images are used.
- A Poisson noise was generated from the data. For example, for the value 10 of pixel 10, the corresponding output pixel is generated from a Poisson distribution with mean 10. A total of 300 test images are used.

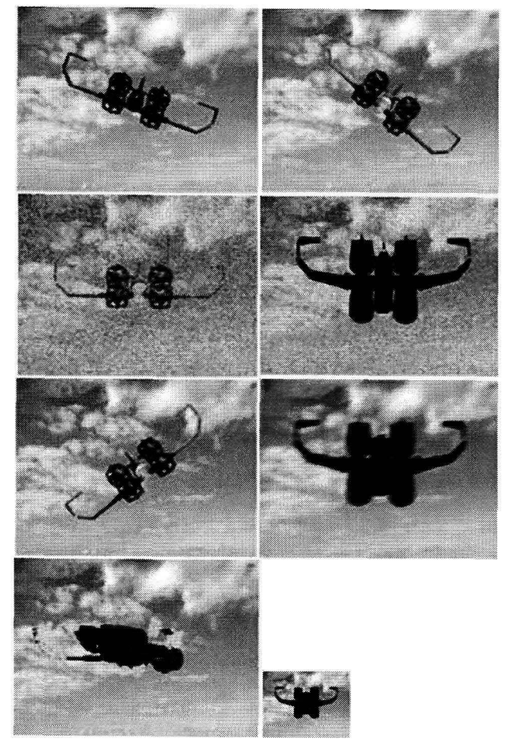

Fig. 8. 1st row: Pose change; 2nd row: Gaussian and speckle noise; 3rd row: Poisson noise and the blurred image; 4th row: occluded image and a scaled down object.

Test 3

Motion blur images are generated by varying the linear motion of a camera by LEN number of pixels, with an angle of THETA degrees in a counter clockwise direction. A motion blur of LEN 1 to 21 with an increment of 2 and THETHA value of 1 to 21 with an increment of 5 has been added to the original images. A total of 300 test images are used.

Test 4

Random occlusions are created on a total of 300 test images with a minimum of 5% and a maximum 50% degree of occlusion.

It is found that for the purpose of object identification, ASIFT and SIFT are more immune to noise and scale changes and recognize objects with 100% of accuracy even under 50% occlusion. Among the moment functions Zernike moments are noise immune, but degrade the performance with the scale increase and the occlusion.

Tables 2, 3 and 4 present the results of testing, where HM – Hu moments, CRM – Complex Radial, ZM – Zernike moments

Table 2. Average object identification accuracy (%).

Type	Scale	Gauss. noise	Pois. noise	Speck. noise	Blur	Occl.
HM	98	62	100	60	100	42
CRM	99	99	100	96	100	48
ZM	99	100	100	97	100	57.5
SIFT	100	100	100	100	100	100
ASIFT	100	100	100	100	100	100

Table 3. Average accuracy (%) of pose estimation under different distortions for database 1.

Type	Scale	Gauss. noise	Pois. noise	Speck. noise	Blur	Occlusion
Hu	78.5	52	89.5	72	69	47
Complex Radial	81	79	86	77	93	60
Zernike	87	81.5	93	80.5	82	54.5

Table 4. Average accuracy (%) of pose estimation under different distortions for database 2.

Type	Scale	Gauss. noise	Pois. noise	Speck. noise	Blur	Occlusion
Hu	85	64.5	98	76.5	77	44
Complex Radial	92.5	87.5	89.5	81	93.5	55
Zernike	96	89	94	87.5	91.5	52

In all the tests on pose estimation, Zernike moments provide the highest accuracy for 6 degree minimal step compared to 4 degree. For the scale changes of 0.7 to 1.2 without any distortions, Zernike moments provide 100% accuracy. Although 100% of the accuracy is not attained under different image distortions applicable to realistic environments, if there is no abrupt scale change, which is the case with the flying object such as an UAV, the approach allows for continuous tracking and pose adjustment. Thus, Zernike moments are selected for hardware implementation of object identification and pose estimation.

4. Hardware implementation

The DM642 Evaluation Module (EVM) [20] is a low-cost standalone development platform that enables users to evaluate and develop applications for the TI C64xx DSP family. The EVM also serves as a hardware reference design for the TMS320DM642 and creates a TI Code Composer Studio DSP/BIOS project to compile, execute and run the generated code on the target processor.

The Simulink models are composed of several blocks from the Simulink library, the C6000lib libraries, and from the Signal Processing and the Video and Image Processing (VIP) blocksets. The C6000lib libraries form part of the Embedded Target for TI C6000 DSP, and contain the DM642 EVM video capture and display routines. The Signal Processing and VIP blocksets provide block libraries for many signal and image processing operations.

More complex algorithms, which are not available in the Signal Processing and VIP blocksets, can be composed from Embedded MATLAB block. This enables the code to be produced, which can be easily profiled and optimized in the Code Composer Studio Integrated Development Environment (IDE).

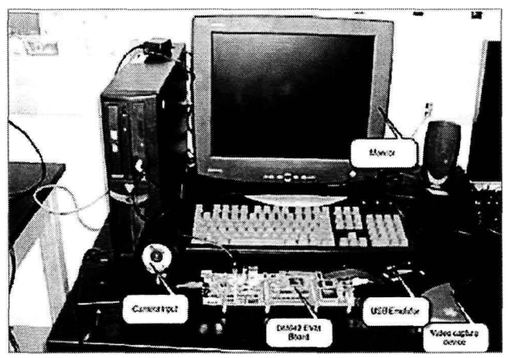

Fig. 9. Experimental setup

We have designed and constructed an Embedded Development System using the DM642 (Fig. 9). This consists of a CCD camera, monitor display, video capture device, XDS510 usb emulator and DM642 EVM, together with a Lithium Ion battery and suitable power conversion circuitry. The EVM contains a 720 MHz TMS320DM642 DSP [6], which has 16KB L1P and L1D program and data cache memories, 256KB configurable L2 cache/mapped memory, 32MB of external SDRAM and 4MB of non-volatile flash memory.

The algorithm has been implemented using MATLAB R2006a, and re-designed in Simulink using MATLAB R2007b, with Code Composer Studio v3.3 to "fit it" to on-board implementation.

For the system, the base sample time is 200 ms and the CPU clock speed is 600 MHz. A much better solution to improve execution times is to use MATLAB's built-in code profiler. Profile Report summarizes the maximum and average times for each subsystem.

Table 5 compares the execution time of our implementations on the PC (2.0 GHz Intel core 2 processor with 2GB of memory) and DM642 EVM board.

Table 5. Execution time comparison.

Algorithm	Average execution time per frame	
	PC	DM642 EVM
Background subtraction	355.91 ms	28.11 ms
Detection	287.23 ms	15.37 ms
Tracking	42.28 ms	1.28 ms
Pose estimation	327.11 ms	22.87 ms

Speedup = ST/st = 14.97, where ST is the sum of execution times of PC and st is the sum of execution times of DM642 EVM. Thus, the average total execution time is 67.3 ms per frame. That yields 14 frames per second, which provides real time processing for the application.

5. Conclusion

This paper has presented a solution for coordinated fleet flying through continuous tracking of the leading object in video sequence and readjusting the flight parameters of the "slave" object accordingly. We have developed a system that is able to recognize in real time an object of interest and record the changes in the yaw, roll and pitch angles as small as 6 degrees and distances proportional to the scale change from 0.3 to 1.4. Zernike moment system is used for this purpose. Finally, the solution has been implemented in hardware using Texas Instrument TMS320DM642 EVM board, yielding 14 frames per second which complies with the application needs.

References

[1] Dudani S.A., Breeding K.J., McGhee R.B., *Aircraft identification by moment invariants*, IEEE Trans. Computer., C-26, 1977, 39–46.

[2] Breitenstein M.D., Kuettel D., Weise T., van Gool L., Pfister H., *Real-Time Face Pose Estimation from Single Range Images*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08).

[3] Osadchy M., *Synergistic face detection and pose estimation with energy-based models*, Journal of MachineLearning Research, Vol. 8, 2007, 1197–1215.

[4] Yuan C., Niemann H., *Object Localization in 2D Images based on Kohonen's Self-Organization Feature Map*, IJCNN99 International Joint Conference on Neural Networks Proceedings, Vol. 5, 1999.

[5] Chang K.Y., Ghosh J., *Three-Dimensional Model-Based Object Recognition and Pose Estimation Using Probabilistic Principal Surfaces*, SPIE: Applications of Artificial Neural Networks in Image Processing V, January 2000, 192–203.

[6] Mcivor A.M., *Background Subtraction techniques*, IVCNZ00, Hamilton, New Zealand, November 2000.

[7] Zhiqiang Hou, Chongzhao, *A Background Reconstruction Algorithm based on Pixel Intensity Classification in Remote Video Surveillance System*, National Key Fundamental Research (973) Program in China.

[8] Piccardi M., *Background subtraction techniques: a review*, IEEE International Conference, Vol. 4, SMC 2004, 3099–3104.

[9] McFarlane N., Schofield C., *Segmentation and tracking of piglets in images*, Machine Vision and Applications, 1995, 187–193.

[10] Friedman N., Russell S., *Image segmentation in video sequences: A probabilistic approach*, Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-97), Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1997, 175–181.

[11] Teague M.R., *Image analysis via the general theory of moments*, J. Opt. Soc. Am., 70, 1980, 920–930.

[12] Lowe D.G., *Object recognition from local scale-invariant features*, International Conference on Computer Vision, Corfu Greece, 1999, 1150–1157,

[13] Morel J.M., Yu G., *ASIFT: A New Framework for Fully Affine Invariant Image Comparison*, SIAM Journal on Imaging Sciences, Vol. 2, Issue 2, 2009.

[14] Jeong M.H., You B.J., Lee W.H., *Color region tracking against brightness changes*, Proc. Australian Joint Conference on Artificial Intelligence, Hobart, Australia, Vol. 4304, December 2006, 536–545

[15] Wang J., Yagi Y., *Integrating shape and color features for adaptive real-time object tracking*, Proc. International Conference on Robotics and Biometrics, Kunming, China, December 2006, 1–6,

[16] Comaniciu D., Meer P., *Mean shift: A robust approach toward feature space analysis*, PAMI, 24(5), 2002, 603–619

[17] Zivkovic Z., Krose B., *An EM-like algorithm for color-histogram-based object tracking*, IEEE Conference on Computer Vision and Pattern Recognition, 2004.

[18] Harris C., Stephens M.J., *A combined corner and edge detector*, Alvey Vision Conference, 1988, 147–152.

[19] Kilthau S.L., Drew M.S., Moller T., *Full search content independent block matching based on the fast Fourier transform*, in: Proceedings of IEEE International Conference on Image Processing, 2002, 669–672.

[20] Texas Instruments, *TMS320DM642 Evaluation Module Reference Technical*, August 2003.

[21] Dudani S.A., *Moment methods for the identification of three-dimensional objects from optical images*, M.Sc. thesis, Ohio State Univ., Columbus, OH, 1971.

[22] Mukundan R., Ramakrishnan K.R., *Moment Functions in Image Analysis – Theory and Application*, World Scientific, Singapore, 1998.

[23] Cutler R., Davis L., *View-based detection*, Proceedings Fourteenth International Conference on Pattern Recognition, Brisbane, Australia, 1, Aug. 1998, 495–500.

[24] Cucchiara R., Piccardi M., Prati A., *Detecting moving objects, ghosts, and shadows in video streams*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 25 October 2003, 1337–1342.

[25] Gloyer B., H Aghajan., Siu K.Y., Kailath T., *Video-based freeway monitoring system using recursive vehicle tracking*, Proceedings of SPIE, 2421, February 1995, 173–180.

[26] Zhou Q., Agarwal J., *Tracking and classifying moving objects from videos*, Proceedings of IEEE Workshop on Performance Evaluation of Tracking and Surveillance, 2001.

Received February 24, 2010