An autonomous system for helicopter hover stabilization using video images

Vygolov O.V., Gorbatsevich V.S.*, Kniaz V.V.*, Kashirkin S.V.*, Maiorov V.P.*, Vizilter Y.V.*, Zheltov S.Y.*, Brondz D.S.***

**State Research Institute of Aviation Systems (FGUP «GosNIIAS»), Moscow, Russian Federation*

o.vygolov@gosniias.ru

***Special Design and Technological Bureau «Omega» (FGUP OKTB «Omega»), Veliky Novgorod, Russian Federation*

david.brondz@oktb-omega.ru

Abstract

The paper addresses a project aimed at developing a computer vision system for helicopter hover stabilization using an on-board down looking camera. An implemented approach is based on an innovative algorithm for interframe displacement detection that provides information for a control system to compensate the helicopter drifting. The algorithm performs feature-based matching of video frames by employing Difference of Gaussians filter and Local Binary Patterns descriptor which are modified to ensure high performance and robustness to affine transformations. The results of a computer simulation are shown. The possibility of implementing the algorithm on an FPGA is discussed.

1. Introduction

Autonomous hovering control for rotary-wing aircraft (rotorcraft) is highly demanded nowadays. Being a flight mode unique to rotary-wing aircraft, hovering makes a helicopter the single solution for a variety of challenging applications such as agricultural spraying, rescue operations and wildfire fighting. During the basic hovering a helicopter maintains a desired position and altitude with respect to the ground. Despite the seeming simplicity, pilots often consider hovering among the most challenging aspects of helicopter flight. The main difficulties arise from highly nonlinear nature of helicopter dynamics. Being underactuated systems, helicopters have significant dynamic coupling that is attributed to the force and torque generation process [1]. Moreover helicopter's state could not be fully observed using inertial navigation or GPS data. As a result a vision-based solution can become an additional valuable source of information for hovering control loop.

Vision-based technology for both manned and unmanned helicopters are rapidly developing in recent years. Kang and Ishimatsu have proposed a base outline of vision-based hovering control for a model helicopter [2]. A CCD camera was mounted on a helicopter and a data processor was placed on the ground. The estimation of helicopter's 3D-position was done using feature points of an 'H' landmark placed on the ground. Such approach however requires a priori knowledge of feature points positions in the Earth coordinate system. At the same time Amidi et. al. [3] proposed a visual odometer to estimate helicopter's state by tracking of ground features.

A simultaneous localization and mapping (SLAM) [4] algorithms were used in a number of projects focused on control of unmanned aerial vehicles (UAV) to provide a solution working for arbitrary hovering location. Brockers et. al. [5] presented a fully self-contained system for navigation and landing of unmanned micro aerial vehicles (MAV). The system uses images from a monocular camera to estimate the state of MAV using SLAM algorithm. The state estimation from SLAM is then fused with the data from an inertial measurement unit (IMU) to provide robustness against outliers.

The quality of SLAM-based helicopter state estimation is highly dependent on feature extraction and matching algorithm. Even though there are a number of robust descriptors that solving this problem (SIFT [6], SURF [7], ORB [8] and others), their implementation for on-board real-time systems is quite challenging task due to computational complexity. Also some of these descriptors [8] demand training, which is not always possible in practical tasks.

In this paper a solution is proposed for helicopter hover stabilization that employs an innovative algorithm for interframe displacement detection by using fast hardware-supported feature extraction and matching techniques designed for an on-board implementation.

The rest of the paper is organized as follows. In Section 2 a brief overview of the developing computer vision system is given, including the system architecture and main hardware components. In Section 3, the interframe displacement detection algorithm is proposed and its potential hardware implementation is discussed. The examples of computer simulation are shown in Section 4. The paper is concluded with the brief plan of further works in the near future.

2. Computer vision system overview

A computer vision (CV) system is being developed to detect a helicopter displacement from a hovering position by using an on-board down looking camera. Information from the CV system is further employed in control signals to compensate the helicopter drifting in a hovering mode. It is assumed that the helicopter can make smooth ascending or descending manoeuvres or change the altitude abruptly in the range of ±5 meters. Allowed roll and pitch ranges are ± 5°. A laser rangefinder is placed coaxially with the camera principal axis. The data from the rangefinder is used to estimate the current altitude of the helicopter.
The CV system architecture is shown schematically on Figure 1.

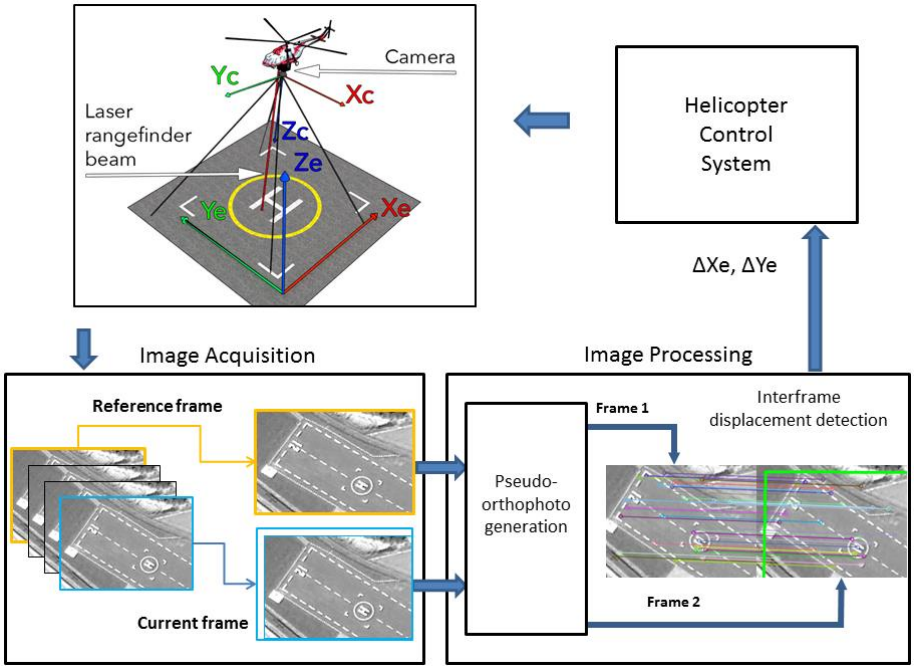

Figure 1: Architecture of the computer vision system for helicopter hover stabilization

A sensor and a platform are developed by FGUP OKTB “Omega” (Russian Federation). Figure 2 shows the developed KT-27 camera intended for an on-board installation. The main specifications of the camera are given in Table 1. The platform has a scalable architecture and includes a number of processing modules (Figure 3) that can share camera control and image processing tasks.

Figure 2: KT-27 camera

Table 1: KT-27 camera specifications

Parameter	Value
Pixel resolution	1024x768 pixels
Dynamic resolution	16 bpp
Field of view	$5{,}4^0$ to 50^0, optical zoom
Sensitivity	1 to 100000 Lux
Frame rate	15 fps
Digital data interface	FC – PH / 25 MBps
Control channel	ARINC – 429
Operation temperature	- 60 + 70 ^{0}C
Consumption	8 W (+ 27 V)
Weight	1500 g
Dimensions	123x75x135 mm

Figure 3: Processing module

A working principle of the CV system is as follows:
1. Capture a reference image at the start of a hovering mode;
2. Estimate displacement between the current image in the hovering mode and the reference image;
3. Send estimated values along X_e and Y_e axes to a helicopter control system to compensate the helicopter drifting.
4. Wait a command from the control system for capturing another reference image.

In this paper the focus is put on the algorithm for interframe displacement detection, which is described in the next section in more details.

3. Algorithm for interframe displacement detection

In this paper the problem of interframe displacement detection is proposed to be considered as a more general problem of finding transformation parameters from one frame of a video sequence to another. An implemented approach is related to a group of methods that set correspondence between two images based on feature extraction and matching techniques. It should be noted that compared to [9] there is no direct reference to the type of interframe transformation matrix which describes the motion model. Therefore the proposed algorithm can be applied to more complex patterns of motion type.

The proposed algorithm includes the following steps:

1. Pseudo-orthophoto generation for reduction of the interframe displacement estimation to 2D-problem;
2. Feature points detection;
3. Feature matching;
4. Creating hypothesis for camera movement.

3.1 Pseudo-orthophoto generation

For helicopter hovering control a 3D-shift and rotation estimation problem could be reduced to a 2D-shift estimation problem. This can be achieved by generation of an orthophoto – orthographic projection of captured frames to a reference plane. Then the shift of the helicopter along X_e and Y_e axes could be measured directly from the generated orthophoto. The generation of orthophoto is done using plane rectification [10].
As a real digital elevation map of the observed terrain, that is required for generation of the orthophoto, is unavailable, it is proposed to approximate observed terrain with a mean level plane. The plane passes through the point of intersection of the laser rangerfinder beam and the observed terrain and coplanar with local horizon plane. A pseudo-orthophoto is generated by projection of the captured frame to the mean level plane. The height of the helicopter above the mean level plane is estimated using laser rangerfinder and IMU data. Camera pitch and roll required for orthophoto generation are also provided by the helictoper's IMU.
It can be shown that the total error in the position of an observed point Δe is proportional to an angular distance of the point from optical center of the frame φ and distance Δh between approximated plane and true position of the point:

$$\begin{aligned} \Delta e_x &= \Delta h \cdot \tan(\gamma + \varphi) \\ \Delta e_y &= \Delta h \cdot \tan(\theta + \varphi) \\ \Delta e &= \sqrt{\left(\Delta e_x{}^2 + \Delta e_y{}^2\right)} \end{aligned} \qquad \textbf{(1)}$$

where Δe_x – error along *Xe* axes, Δe_y – error along *Ye* axes, γ – camera roll angle, θ – camera pitch angle (Figure 4).

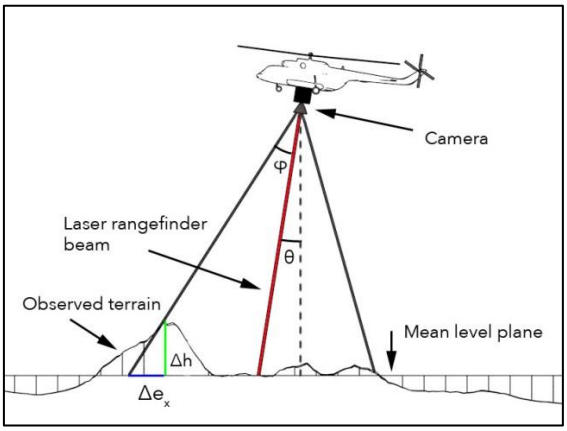

Figure 4: Illustration of error-terrain dependency

If the camera's FOV is reduced then the value of φ will tend to zero, and the total error will also tend to zero. As a result the wide FOV of 50° is used for initial feature point detection. For precise stabilization mode a narrower FOV is used with φ limited to ±5° by either optical zoom or automatic selection of region of interest in the centre of the frame. For such configuration the average value of Δe doesn't exceed 1 m for a helicopter hovering under a rugged terrain with average Δh of 10 m.

3.2 Feature points detection

While selecting a certain feature detection algorithm, the special attention was paid to two main factors:
- high overall performance of the algorithm;
- high stability to affine and projective distortions.

As a result the DoG algorithm [11] has been selected as the most suitable basis for the proposed feature detector. DoG provides feature points which are the local maxima of the response function:

$$\boldsymbol{DOG_k(x,y)} = \left| \boldsymbol{I} * \frac{\boldsymbol{1}}{\boldsymbol{2\pi\sigma_1^2}} \boldsymbol{e}^{\frac{-\left(x^2+y^2\right)}{2\sigma_1^2}} - \boldsymbol{I} * \frac{\boldsymbol{1}}{\boldsymbol{2\pi\sigma_2^2}} \boldsymbol{e}^{\frac{-\left(x^2+y^2\right)}{2\sigma_2^2}} \right| \quad \textbf{(2)}$$

Where

I – input image;
σ_1, σ_2 – algorithm parameters;
x, y – coordinates of the point of interest;
k – identifier of the pair $\{\sigma_1, \sigma_2\}$;
* – convolution operation.

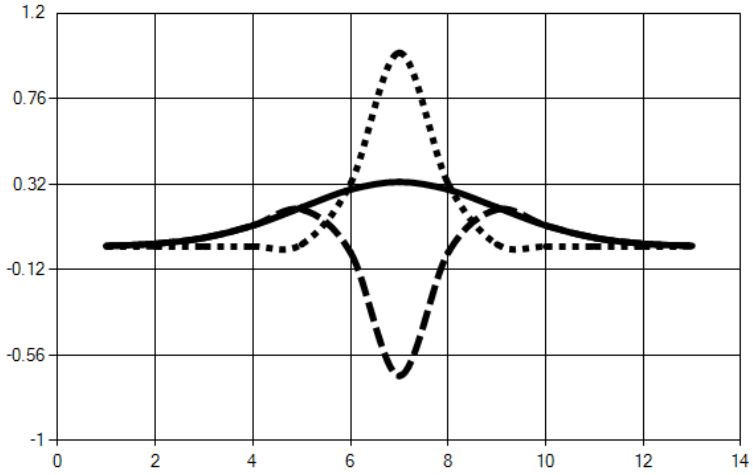

Figure 5: IRF of the 1-D DoG filter (lower dashed curve)

Those feature points are robust to all distortions occurring from the affine transformations except the change of a scale. This drawback is proposed to reduce by detecting feature points for different pairs of σ_1, σ_2 parameters. To cope with computational load, related to the image convolution with different Gaussian kernels, it is suggested using the particular type of recurrent filters [12].
At the same time, from a practical point of view, it is possible to apply a simple recursive first order filter to accelerate computation. The form of the filter is:

$$H(z) = \boldsymbol{H_R} * \boldsymbol{H_L}, \quad \textbf{(3)}$$

$$\boldsymbol{H_R} = \frac{\boldsymbol{A}}{\boldsymbol{b_0 - b_1 z^1}}, \boldsymbol{H_L} = \frac{\boldsymbol{A}}{\boldsymbol{b_0 - b_1 z^{-1}}}, \quad \textbf{(4)}$$

where b_0, b_1 – filter coefficients.
The filter coefficients are chosen empirically to minimize the residuals: the square of the difference between the values of IRF of recurrent filter and true values of approximated Gaussian function.

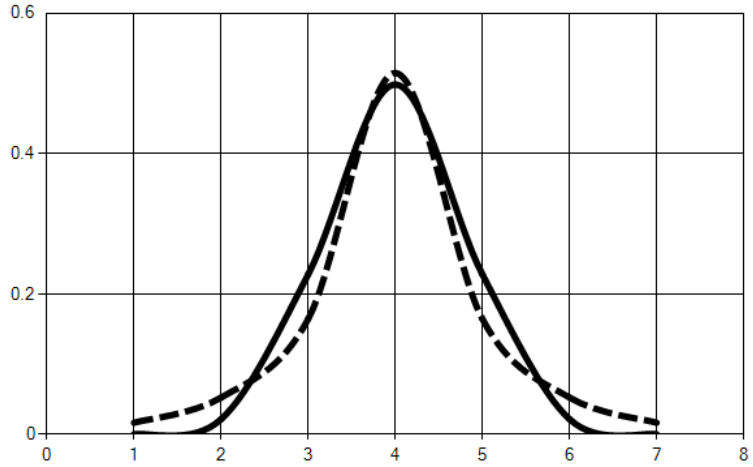

Figure 6: IRF of a first order recurrent filter (dashed) and a Gaussian function (solid)

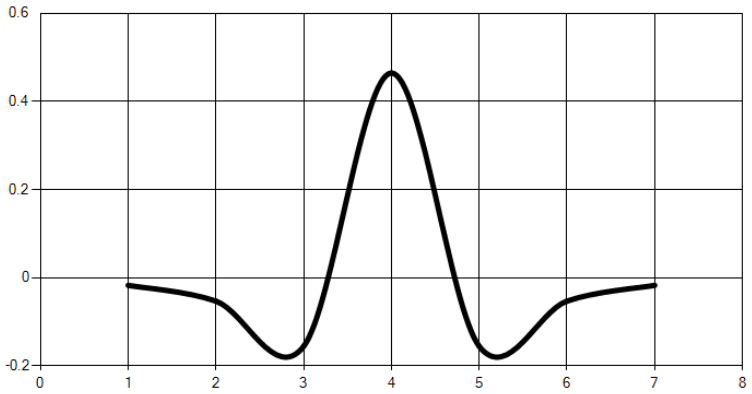

Figure 7: IRF of the first order recurrent filter is used directly for feature point detection

As a result the proposed approach provides a very fast detection of feature points with almost identical stability compared with the direct method of convolution calculation.

3.3 Feature matching

The next step in the algorithm workflow is feature matching, that is to say finding pairs of feature points from two images of a video sequence which correspond to the same point of a scene. In the considered task the assumption can be made that image registration conditions change slightly in intervals between video frames. This allows more simple descriptors to be applied. In this paper, it is proposed to use LBP-descriptors with various radii when comparing image features.

Descriptor computation for this approach can be briefly described as follows: for each point of the image the circular neighbourhood of a certain radius is considered, each point of the neighbourhood is compared to the current point of the image:

$$\boldsymbol{LBP(p)} = \sum_{i=0}^{n} \boldsymbol{2^i} \begin{matrix} \mathbf{1}: \boldsymbol{I(p) \geq I(o_i)} \\ \mathbf{0}: \boldsymbol{I(p) < I(o_i)}' \end{matrix} \quad \textbf{(5)}$$

where

n – number of points of the circular neighbourhood;
o_i – neighbourhood point i;
p – current image point.

The comparison result is stored as a binary vector. The Hamming distance is used for their comparison.

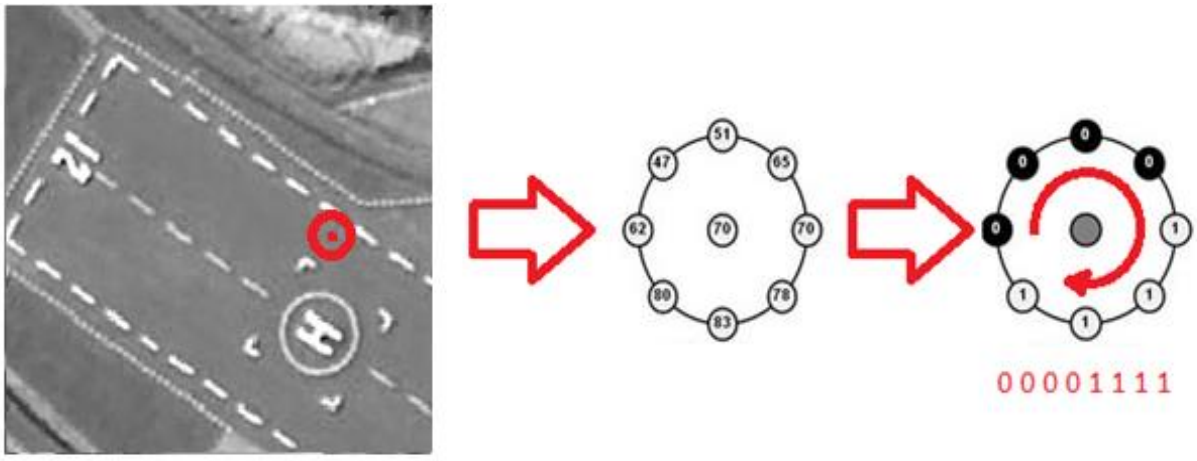

Figure 8: Example of LBP calculation – an image point descriptor

However the direct use of LBP for comparison of pairs of points is difficult because of its high sensitiveness to affine transformations. To make LBP more stable, the DoG features detector responses are used instead of image intensities along with local maxima in some neighbourhood instead of image points:

$$\boldsymbol{MaxLBP_k(p)} = \sum_{i=0}^{n} 2^i \begin{matrix} \mathbf{1}: \boldsymbol{DOG_k(p) \geq MaxDOG_k(o_i)} \\ \mathbf{0}: \boldsymbol{DOG_k(p) < MaxDOG_k(o_i)} \end{matrix} \quad \textbf{(6)}$$

$$\boldsymbol{MaxDOG_k(x,y)} = \boldsymbol{max_{i\in[x-\Delta,x+\Delta],j\in[y-\Delta,y+\Delta]} DOG_k(i,j)}, \quad \textbf{(7)}$$

where

n – number of points of the circular neighborhood;
o_i – neighbourhood point i;
p – current image point;
Δ – radius of the point's neighbourhood.

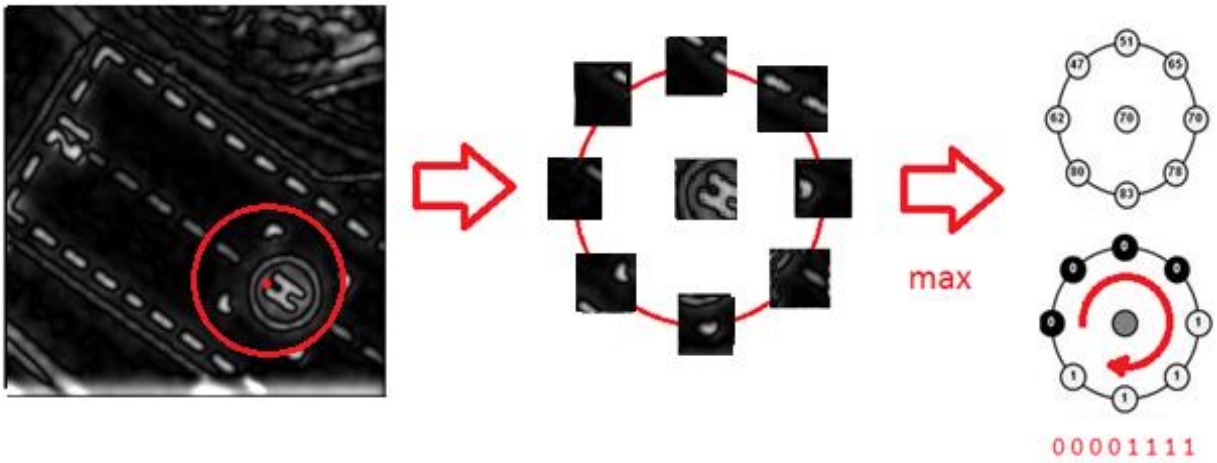

Figure 9: Example of MaxLBP calculation

Proposed descriptor has a big resistance to affine distortions and is fast to compute because of direct calculation with the use of a local maxima map. This local maxima map is calculated at the special point search stage. In addition, using the assumption about small changes of image registration conditions, it is reasonable to add direct comparison of function responses of the feature detector in every point:

$$\boldsymbol{D(a,b) = \sum_{k=1}^{K} D_{Hamming}\left(MaxLBP_k(a), MaxLBP_k(b)\right) + C\,(DOG(a) - DOG(b))^2} \quad \textbf{(8)}$$

where

a, b – compared points;
$D_{Hamming}$ – Hamming Distance;
C – weight coefficient.

It is also expedient to choose rather small coefficient C to ensure the response function difference has an impact only in dealing with "controversial" situations.
Figure 10 shows example of feature matching for pseudo-orthophoto images.

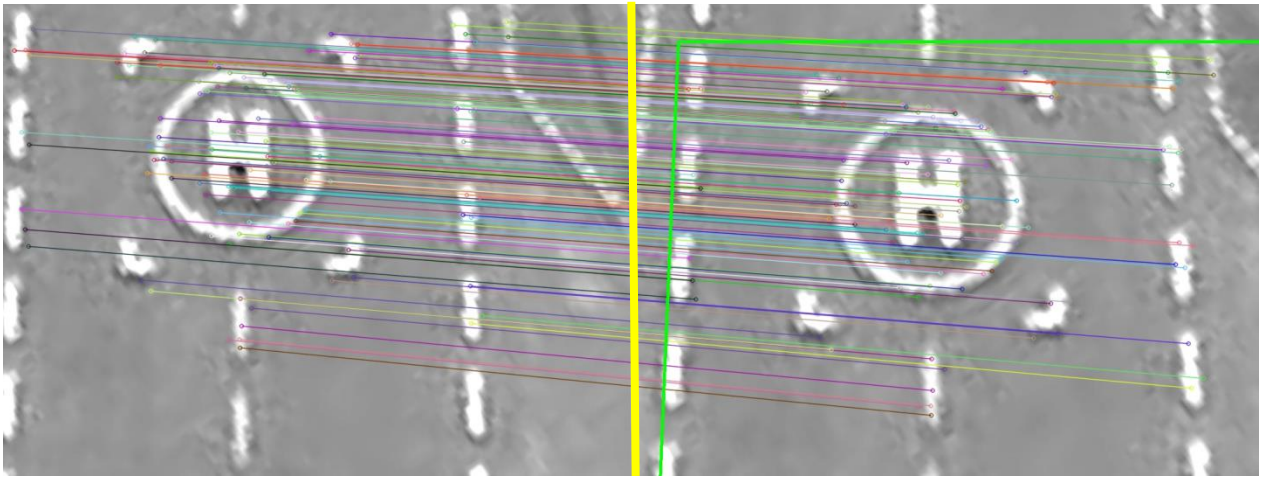

Figure 10: Examples of feature matching: reference frame (left), current frame (right), features correspondence is shown by lines, green coloured frame shows interframe displacement

3.4 Creating hypothesis for camera movement

One of the most popular algorithms for selecting the matrix parameters for affine or projective transformations is RANSAC [13]. Unlike least square based methods RANSAC much better handles a large number of significant occasional emissions in the input data. The basic idea of the algorithm is to use random sampling to select the optimal values of the transformation matrix. The algorithm in its classic form consists of the following steps:
Step 1. From the set of pairs of points, randomly allocate n pairs. The value of n should be sufficient for an accurate calculation of the transformation matrix.
Step 2. Create a hypothesis about form of the transformation matrix based on the selected pair from the Step 1.
Step 3. Verify the hypothesis from Step 2 usually by using the following formula:

$$\boldsymbol{Q(A) = \sum_{(x1,x2)\in X} \begin{matrix} \mathbf{1}: (A\,x1 - x2)^2 \le dist \\ \mathbf{0}: (A\,x1 - x2)^2 > dist' \end{matrix}} \quad \textbf{(9)}$$

where

A – transformation matrix;
X – set of pairs of points on adjacent images;
$x1, x2$ – vectors corresponding to the coordinates of the image points;
$dist$ – threshold, defining the maximum point noise deviation.

The algorithm stops when one is true: the hypothesis is generated with sufficient quality assessed by the functional $Q(\cdot)$ or a certain iterations number is achieved.
The drawback of this approach to transformation matrix calculation is its extremely high sensitivity to the selected threshold *dist*. In this study, a modified functional of quality is proposed:

$$\boldsymbol{Q^*(A) = \sum_{(x1,x2)\in X} \begin{matrix} \mathbf{1}: (A\,x1 - x2)^2 \le dist \\ \mathbf{0}: (A\,x1 - x2)^2 > dist \end{matrix} + K \sum_{(x1,x2)\in X} (A\,x1 - x2)^2,} \quad \textbf{(10)}$$

where K is chosen sufficiently small to meet the requirement:

$$K \sum_{(x1,x2)\in X} (A\,x1 - x2)^2 \leq 1. \quad (11)$$

Usage of such functional allows accurately resolving "controversial" situations in which two hypotheses receive the same number of votes.

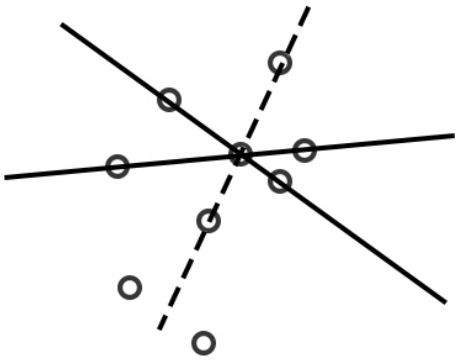

Figure 11: Example of "controversial" situation in the task of finding a straight line. It can be seen that despite the fact that each hypothesis has an equal number of votes, the dashed line is the most probable candidate since there are two extra points in its neighbourhood

3.5 Discussion of a potential hardware implementation

A hardware implementation of the developed algorithm is a crucial task for the completion of the whole computer vision system due to the following factors. On the one hand, an on-board platform is usually quite limited in computational resources. On the other hand, hover stabilization task requires the real-time processing of a big amount of video data. To solve this problem the hardware implementation approach is proposed that combines an FPGA and a CPU. Such combination gives possibility to process a high resolution video stream in real-time mode using the FPGA and, at the same time, preserves flexibility in choosing the data processing procedures for the algorithm provided by the CPU. Typical dataflow for the approach is shown on Figure 12.

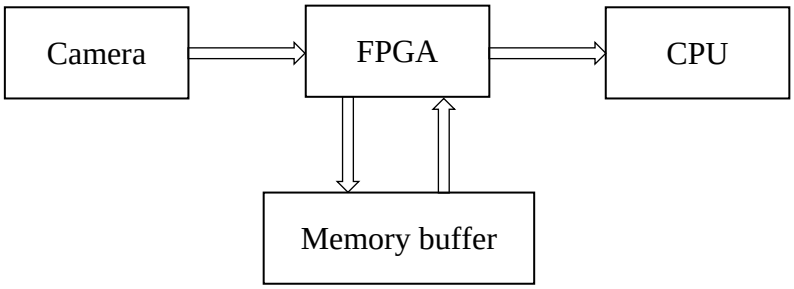

Figure 12: Proposed data flow between the camera, FPGA, memory and CPU

The proposed approach will be implemented using a processing board based on the Arriva V GX FPGA family (see Figure 13) which is suitable for complex multistage data processing. The algorithm is structured in such a way that the FPGA takes steps with a big data flow or regular processing procedures (e.g. an image acquisition, Gaussian convolution, search of local maxima). For the camera control and decision making tasks a CPU with ARM architecture will be used.

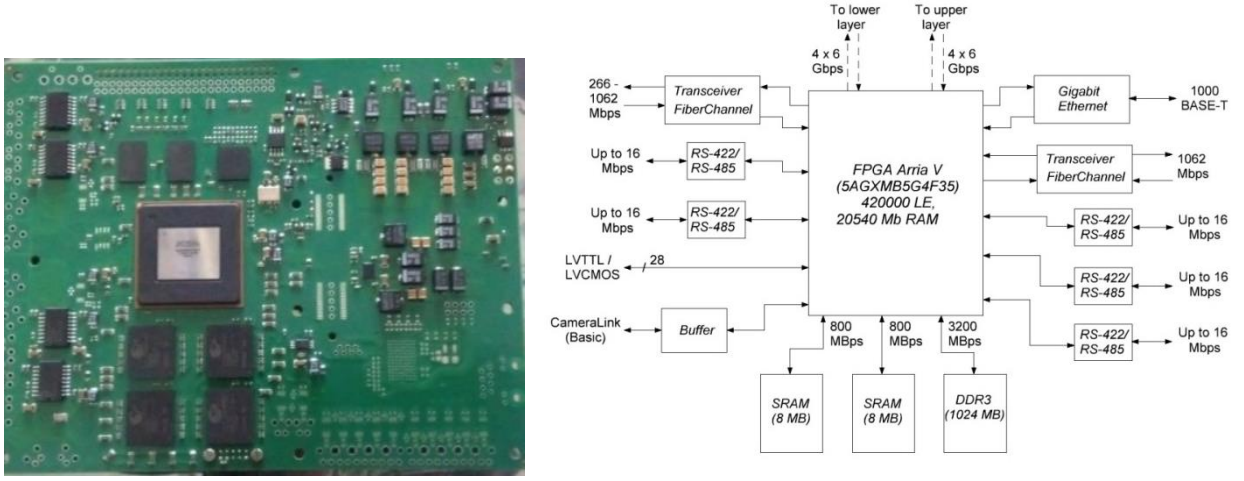

Figure 13: The processing board with FPGA (left) and its structure (right)

Preliminary performance tests have shown operability of the proposed approach with a potentially more than twice reduction in CPU load by using video stream processing on the FPGA.

4. Computer simulation

Experimental results are currently being obtained by a computer simulation using a developed External Data Simulation System (EDSS). The EDSS is a highly realistic real-time simulator that provides a view out of the cockpit saturated by three-dimensional (3D) models of objects and landscape along with imitation of optical sensors outputs. The EDSS includes the following main components to generate such a virtual environment:

- 3D models of terrain and infrastructure;
- Dynamic models of aerial and ground objects;
- Textures of the terrain and objects for KT-27 camera view;
- Simulation software for weather conditions, time of day, atmosphere conditions, sky textures, special effects;
- Simulation software for scenarios and processes, including an implementation of a helicopter flight dynamics model.

Figure 14 shows examples of 3D-scences generated by the EDSS. Figure 15 shows examples of feature matching on simulated images generated by the EDSS.

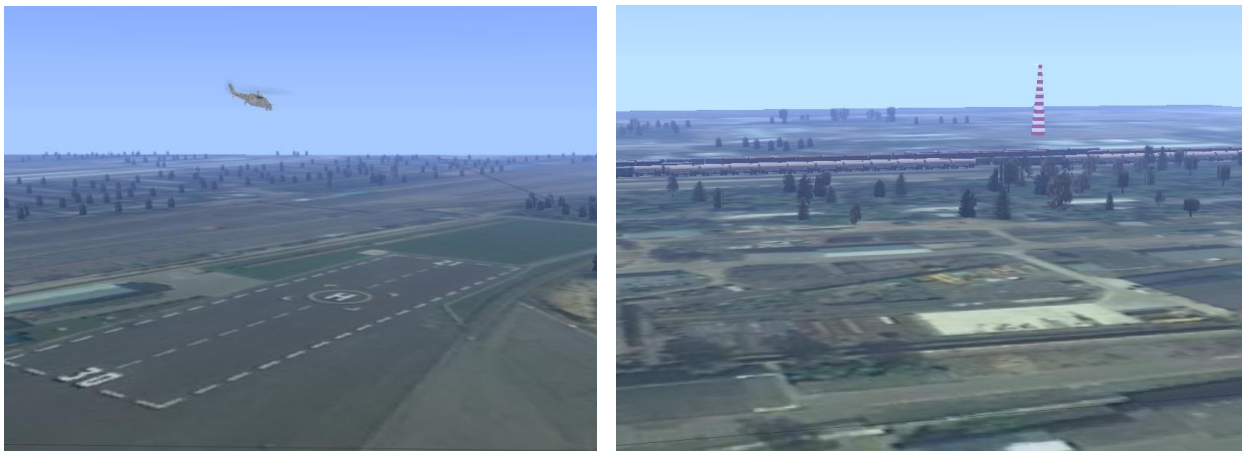

Figure 14: Examples of 3D-scences generated by the EDSS:
helipad in mist (left); field landscape (right)

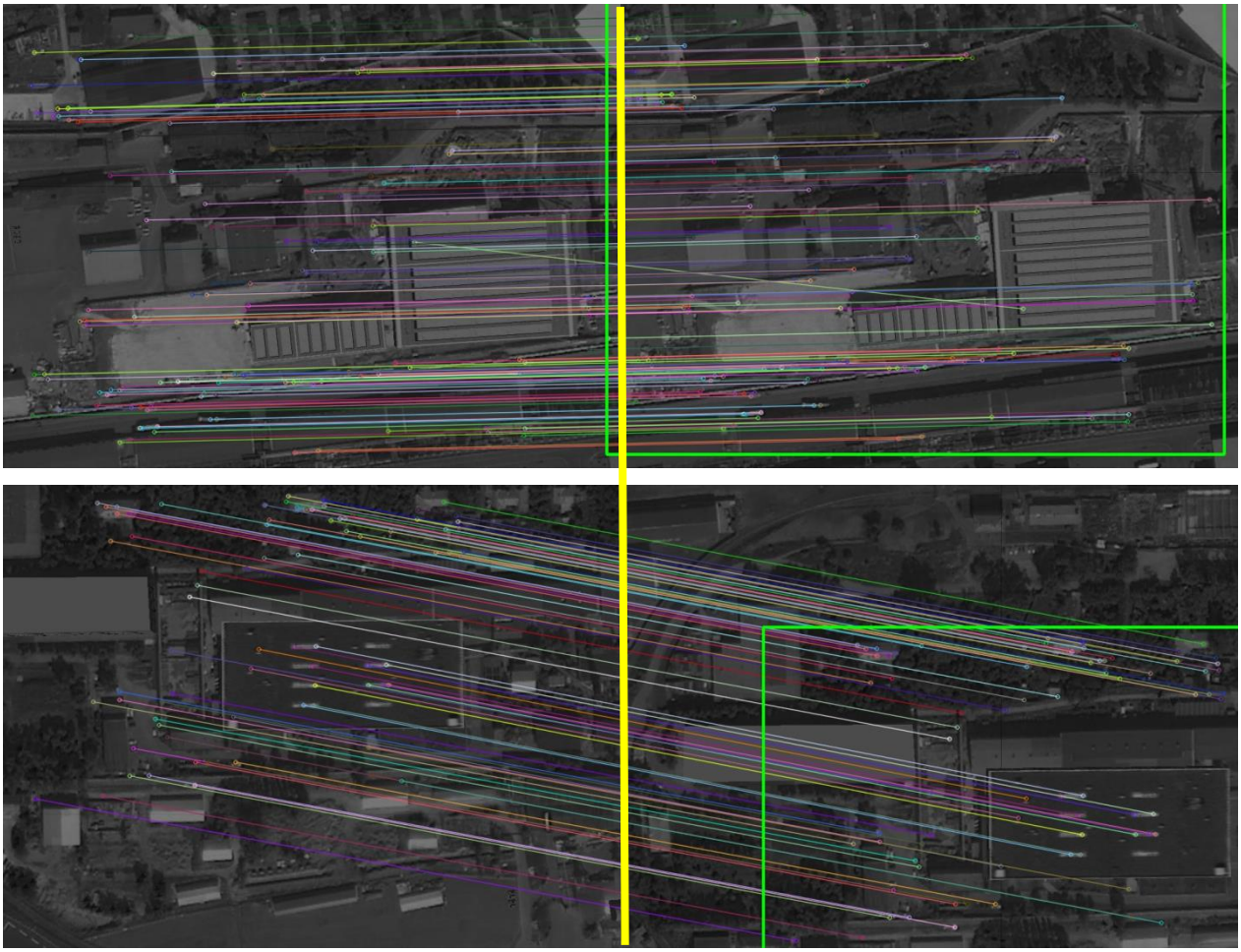

Figure 15: Examples of feature matching on simulated images of an urbane landscape generated by the EDSS: reference frames (left column), current frames (right column), features correspondence is shown by lines, green coloured frame shows interframe displacement

5. Conclusions and plans

In this paper the possibility of vision-based helicopter hover stabilization is studied. A computer vision system is being developed that includes a downward looking camera, a laser rangefinder placed coaxially with the camera principal axis, and a scalable platform which shares camera control and image processing tasks between a CPU and an FPGA circuit. An innovative algorithm for interframe displacement detection is proposed, which introduces the fast modifications of DoG filter and LBP descriptor designed for a real-time hardware implementation in low-powered on-board systems. Some ground based experiments were performed using simulated video data that reached a conclusion that the proposed algorithm can be a perspective solution for a helicopter drifting detection in a hovering mode.

The major tasks that remain to complete in the near future are the following: hardware implementation of the interframe displacement detection algorithm, the platform integration to the EDSS and its ground based tests using simulated video data, flight experiments with the KT-27 camera for real video data acquisition, ground based tests of the system using real video data.

References

[1] Alvarenga J., Vitzilaios N. I., Valavanis K. P., Rutherford M. J. 2014. Survey of Unmanned Helicopter Model-Based Navigation and Control Techniques. *Journal of Intelligent & Robotic Systems.* 1–52.

[2] Kang, C.-U., Ishimatsu T., Shimomoto Y., Satake J. 1999. Hovering control of model helicopter by vision. *Journal Artificial Life and Robotics*.197–201.

[3] Amidi O., Kanade T., and Fujita K., 1999. A Visual Odometer for Autonomous Helicopter Flight. *Robotics and Autonomous Systems*, 28, 185–193.
[4] Davison A. J., 1999. Mobile Robot Navigation Using Active Vision.
[5] Brockers R., Susca S., Zhu D., Matthies L. 2012. Fully Self-Contained Vision-Aided Navigation and Landing of a Micro Air Vehicle Independent from External Sensor Inputs. *Proc. SPIE 8387, Unmanned Systems Technology XIV*.
[6] Lowe D. G., 1999. Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision. doi:10.1109/ICCV.1999.790410 2*, 1150–1157
[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features," *Computer* Vision and Image Understanding (CVIU) vol. 110, pp. 346–359 (2008)
[8] Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski "ORB: an efficient alternative to SIFT or SURF," Computer Vision (ICCV) IEEE International Conference on., pp. 2564–2571 (2011)Martin A. Fischler, Robert C. Bolles "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Comm. Of the ACM 24, pp. 381–395 (1981)
[9] Boris Vishnyakov, Vladimir Gorbatsevich "Fast algorithm for background matching and digital stabilization of shaking or moving camera," Technical Vision in Computer Systems conference thesis, in Russian. (2013)
[10] Luhman T., Robson S., Kyle S., Boehm J., 2014, Close-Range Photogrammetry and 3D Imaging, 2nd edition, ISBN 978-3-11-030269-1.
[11] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision 60 doi:10.1023/B:VISI.0000029664.99615.94 2, pp. 91–110 (2004)
[12] Ian T. Young, Lucas J. van Vliet. "Recursive implementation of the Gaussian filter," ELSEVIER Signal Processing 44, pp. 139-151 (1995)
[13] Martin A. Fischler, Robert C. Bolles "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Comm. Of the ACM 24, pp. 381–395 (1981)