Unmanned Aerial Vehicle (UAV): back to base without satellite navigation

Vladislav Blazhko [1], Alexander Kalinovsky [2], Vassili Kovalev [2]
1) Belarusian State University, Minsk
2) United Institute of Informatics Problems of the NAS of Belarus, Minsk
e-mail: mr_plum@mail.ru, gakarak@gmail.com, vassili.kovalev@gmail.com

Abstract: *In this paper you will see research aimed at accuracy improvement of existing Inertial Navigation Systems on UAV with on-board video camera and computer vision methods.*

Keywords**:** UAV, drones, detectors, descriptors.

1. INTRODUCTION

Nowadays UAV's industry evolves in the different spheres of life. Think about the Amazon Prime Air that needs to deliver purchases to customers or those various military drones that can strike or scout, drones for mapping and etc.

The most of the drones have auto-return home function: when GPS is available, drone remembers the exact spot which it took off from. Wherever drone is flying, if its battery is running very low or user decided to call it back then it return right back home.

This function work well under normal conditions. But under military conditions, where it is very simple to drown out the GPS signal or even emit counterfeit signal, we are forced to find another methods to return to home. Also in some regions the GPS signal can be unstable and weak and cause problems when returning home.

Usually drone has Inertial Navigation System (INS) [1], which contain Inertial Measurement Units (IMU) which have angular and linear accelerometers (for changes in position); some IMUs include a gyroscopic element (for maintaining an absolute angular reference). The purpose of INS is calculate via dead reckoning the position, orientation, and velocity (direction and speed of movement) of a moving object without the need for external references.

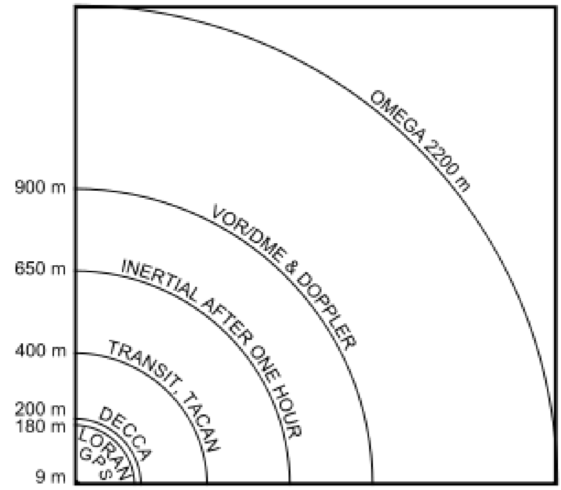

Fig.1 – Accuracy of navigation systems

The main disadvantage of INS is that they have measurement errors and accumulate them. Moreover there is whole class of small drones which cannot afford to equip large or heavy, or expensive INS, it also reduces the accuracy of the measurements.

So if UAV is very far from home and will return only by INS, then it can return to the place which is still far from home (Fig.1).

We conducted several tests with INS on real drone AR.Drone 2.0 (Fig.2) [2]. It has weight about 400 grams and flight time about 15 minutes. Also it has gyroscope with 3 axles, accuracy of 2,000°/second, and accelerometer with 3 axles, accuracy of +/- 50 mg.

Fig.2 – AR.Drone 2.0

We launched this drone from one corner of the room and then sent it to opposite corner through the adjacent corner.

The trajectory which we have obtained from INS presented in Fig.3.

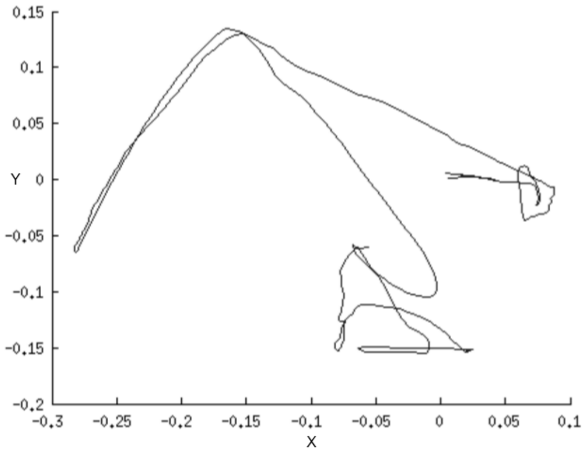

Fig.3 – Trajectory recovered by Inertial Navigate System

As you can see initially the trajectory is good, but over time it gets worse. So we want to clarify the work of INS by developing a small lightweight module, like a smartphone, with camera and processor.

2. MAIN IDEA

The main idea is very simple. While the drone has a signal from satellite navigation system it will take photos of the area on which it flies with some specific period of time. Each photo is linked to physical coordinates. Thus, the drone can represent photos as a trajectory (Fig.4). As soon as the signal of satellite navigation is lost, the drone rises above to easily find the last area on which it flew. And then drone follows the trajectory when returning back.

Fig.4 – Trajectory built on photos linked to coordinates

The find last area is an attempt to find the relevant part of trajectory on the current camera view (Fig.5). If we do this then we can calculate translate, rotate and scale transformation. After we can get approximated coordinates of our location, since the part of trajectory linked to physical coordinates. And finally choose the direction of flight.

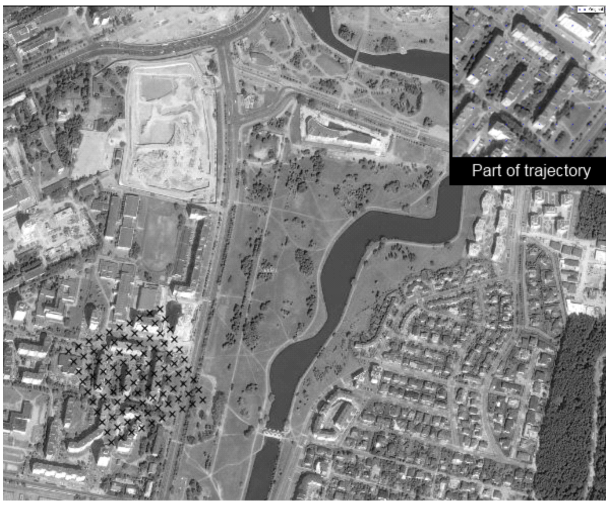

Fig.5 – Find the part of trajectory on current camera view

The trajectory may be very long and finding the relevant part of whole trajectory on the current camera view can require huge computation. Therefore we will use telemetry from INS to define an approximated scope of the trajectory in which we are. It allow us to find the relevant part on the part of trajectory instead of whole trajectory, which in turn reduces the computational complexity.

3. COMPUTER VISION METHODS

The main aim is getting the transformation between current camera view and the relevant part of the trajectory.

More formally the transformation is a matrix of homography, which more correctly as possible transform the points from trajectory to the current camera view. To obtain this matrix we must have a 2D to 2D point correspondences.

So, as a first step, we researched how is good the feature detection, description and matching methods for getting correct transformations. For this purpose we used library OpenCV [3].

Algorithm for getting final transformation presented in Fig.6. We tested follow feature detectors: SIFT, SURF, KAZE, AKAZE, BRISK, ORB, MSER [4-10]. All detectors have optimal parameters.

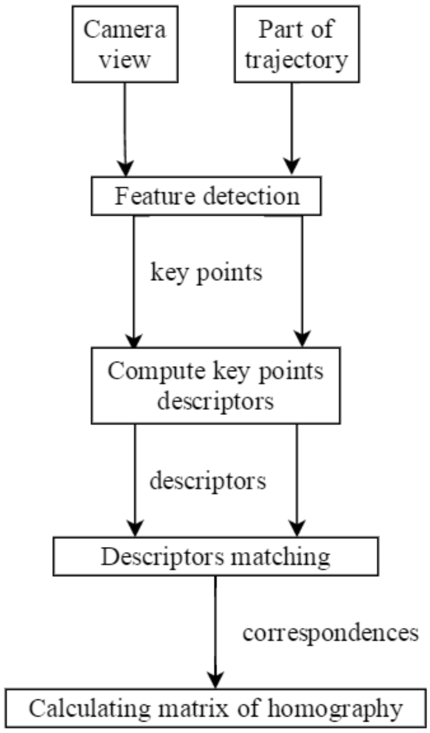

Fig.6 – Algorithm of transformation obtaining

The descriptors same like a detectors plus FREAK [11]. The matcher used BFMatcher (Brute-Force matcher), because we need quality, which takes NORM_L2_SQR [12] for SIFT, SURF, KAZE and AKAZE descriptors and NORM_HAMMING for BRISK, FREAK and ORB descriptors and k equal to 2 (it means, that BFMatcher will search for 2 near neighbours). After this matches ratio tested, it consists in that matches, where the distance between the first and second neighbour point large, are good, because in this case we without doubts can choose between them, otherwise we exclude that match like being able to cause confusion.

For calculating matrix of homography used the function findHomograpy with RANSAC method [13], because not all of the correspodence points fit the rigid perspective transformation.

4. MATERIALS

For our problem we need an algorithm, which will be scale and rotate invariant and fast. So we prepare the dataset, which consist of 4 classes: "no_scale", "scale_1.1", "scale_1.5", "scale_2.0".

Each class contain the main image, which we call map, 49 rotated and scaled (according to the name of class) pieces of this map and 49 matrixes of homography, which perform transformation from map to piece

The map obtained from the program SASPlanet [14], which allow to get the photos from satellite with different zoom like an image. The region of interest was selected and downloaded the map of Yandex satellite maps (Fig.7) and Google satellite maps on zoom 17 (~1.41 meters per pixel). This region was selected, because it has many different classes: forest, river, buildings, roads and wastelands.

Fig.7 – Map from the Yandex satellite maps

The map size is 1126 x 1226 px and the size of each piece is 353 x 353 px. Example of image from class "scale_2.0" in Fig.8.

Fig.8 – Example of image from class "scale_2.0"

The scaling was done artificially by stretching. Each matrix of homography exactly transform the points of the map to points of the piece, because each matrix was built with known center coordinates, rotate angle and scale multiplier.

4. SCORE DISCUSSING

The accuracy of approximated coordinates of drone location directly depend on the accuracy of transformed points.

Grid score. We inflict the uniform grid of N x N points on the piece of the map. And then transform this points on the map by inverse original matrix of homography and inverse of matrix given by algorithm. Point treated as correct if euclidean distance between corresponding points of original and algorithm transforms on map less or equal than some threshold. The score is a percent of correct points. Example of grid score in Fig.9, where the grid is 5 x 5 points, the white points belong to the grid of algorithm and black - original. There are 5 points match with threshold 5 pixels.

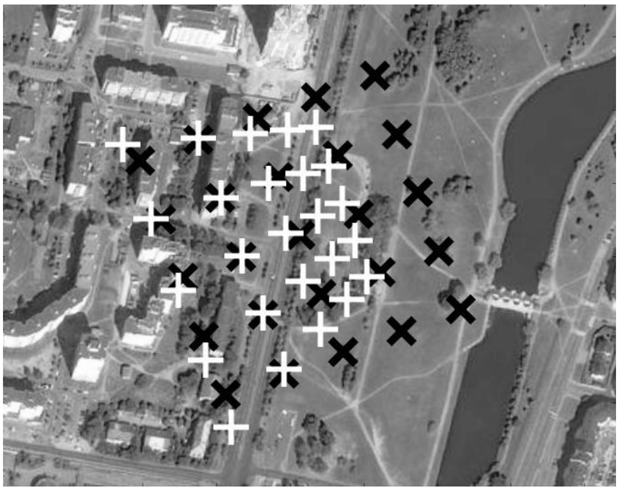

Fig.9 – Example of grid score. Score equal to 0.2

Key points score. This score is approximated score from [15]. We got the matched key points on map and on piece of the map. So we transform matched key points on map to the piece by the original matrix of homography. The score is a percent of correct points.

5. EXPERIMENTAL RESULTS

In our experiment we took the grid 10 x 10 points with threshold equal to 5 pixels. The map from Yandex satellite maps and pieces of map from the same map. Also we have experimented with map from Google satellite maps and pieces of map from Yandex satellite maps, but all algorithms have given almost zero results.

Accumulated grid scores for each class of dataset presented in Fig.10 (more better). Plot consists of 4 layouts, each layout represent the value of grid score for appropriate class. The same plot presented in Fig.11, but of accumulated key points score.

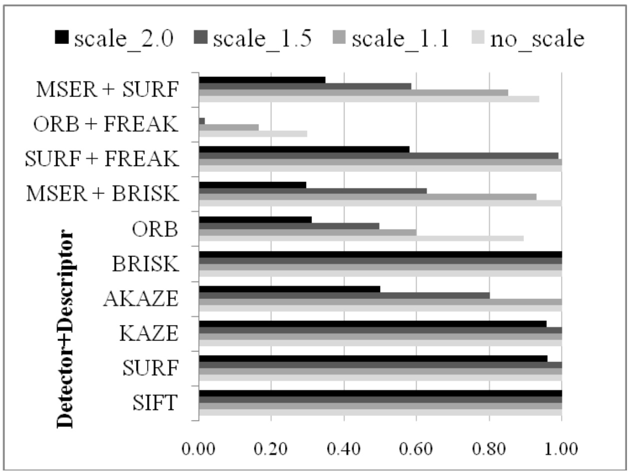

Fig.10 – Plot of accumulated grid score

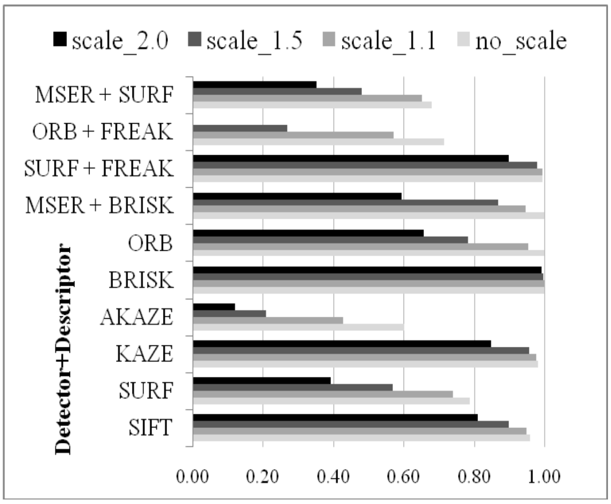

Fig.11 – Plot of accumulated key points score

The average time required for detecting and descripting one key point shown in Fig.12 (less better).

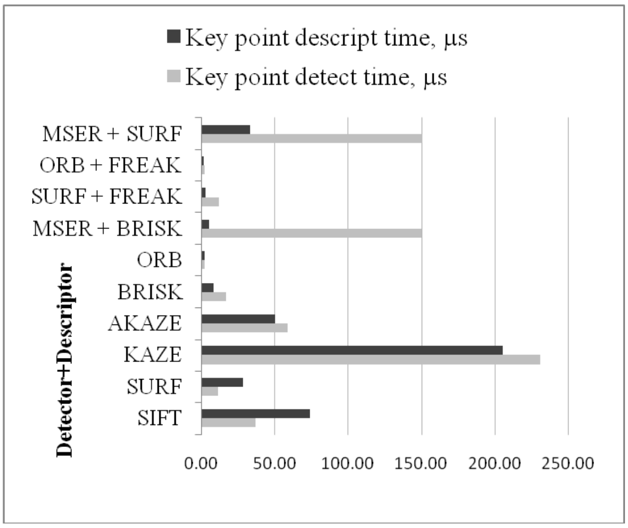

Fig.12 – Plot of time required by detector and descriptor

The link between average full time of calculating homography and average grid score shown in Fig.13. This plot don't display the point corresponding to the ORB with FREAK, because this bunch gave bad results.

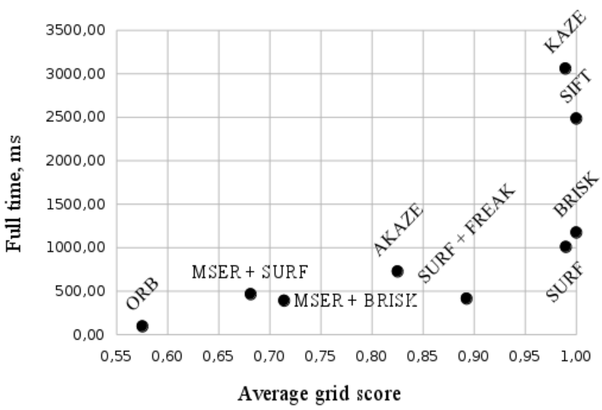

Fig.13 – Scatter plot of average grid score and full time

6. CONCLUSION

As can be seen from the experiment with map from Google satellite maps and pieces of the map from Yandex satellite maps, the approach for accuracy improvement of existing INS can be applied to the drones, which have flight time about several hours. The cause of bad results in this experiment is that the photos from Google and Yandex satellites taken in different seasons and feature detectors select different key points at the same areas.

Another conclusion that can be drawn is that the algorithm, which rely on BRISK detector and descriptor, has a good recovery performance of matrix of homography in both scores and at the same time it has a good time of calculating. Also SURF with FREAK has a good performance in both scores and very little time and can compete ORB.

7. REFERENCES

[1] Kayton, M. (1989) Navigation — Land, Sea, Air, and Space, IEEE Press, New York.

[2] AR.Drone 2.0. Parrot new wi-fi quadricopter - AR.Drone.com - HD Camera - Civil drone - Parrot [Electronic resource]. Mode of access: http://ardrone2.parrot.com/. – Date of access: 13.04.2016.

[3] OpenCV | OpenCV [Electronic resource]. Mode of access: http://opencv.org/. – Date of access: 13.04.2016.

[4] D. Lowe, *Distinctive Image Features from Scale-Invariant Keypoints*, Int'l J. Computer Vision, vol. 2, no. 60, pp. 91-110, 2004.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. J. V. Gool. *Speeded-up robust features (surf)*. CVIU, 110(3), 2008.

[6] P.F. Alcantarilla, A. Bartoli, A.J. Davison: *KAZE features*. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 214–227. Springer, Heidelberg (2012)

[7] P. F. Alcantarilla, J. Nuevo, A. Bartoli, *Fast explicit diffusion for accelerated features in nonlinear scale spaces*, Brit. Mach. Vision Conf.

[8] S. Leutenegger, M. Chli, R. Siegwart. *Brisk: Binary robust invariant scalable keypoints.* In ICCV, 2011.

[9] E. Rublee, V. Rabaud, K. Konolige, G. R. Bradski. *Orb: An efficient alternative to sift or surf*, in ICCV, 2011.

[10] J. Matas, O. Chum, M. Urban, T. Pajdla. *Robust wide baseline stereo from maximally stable extremal regions*, in Proc. of British Machine Vision Conference, p. 384-396, 2002.

[11] A. Alahi, R. Ortiz, and P. Vandergheynst. *Freak: Fast retina keypoint*, in CVPR, 2012

[12] R. Arandjelovic, A. Zisserman (2012). *Three things everyone should know to improve object retrieval*, in CVPR.

[13] M. A. Fischler, R. C. Bolles. *Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography*. Communications of the ACM, 24(6):381–395, June 1981.

[14] SASGIS - Веб-картография и навигация >> SAS.Планета [Electronic resource]. Mode of access: http://sasgis.ru/sasplaneta/. – Date of access: 14.04.2016.

[15] K. Mikolajczyk, C. Schmid. *A performance evaluation of local descriptors*, in PAMI, 2004.