

VISION ANALYSIS SYSTEM FOR AUTONOMOUS LANDING OF MICRO DRONE

Marcin SKOCZYLAS*

*Faculty of Computer Science, Białystok University of Technology, ul. Wiejska 45A, 15-351 Białystok, Poland

marcin.skoczylas@pb.edu.pl

Abstract: This article describes a concept of an autonomous landing system of UAV (Unmanned Aerial Vehicle). This type of device is equipped with the functionality of FPV observation (First Person View) and radio broadcasting of video or image data. The problem is performance of a system of autonomous drone landing in an area with dimensions of $1\text{m} \times 1\text{m}$, based on CCD camera coupled with an image transmission system connected to a base station. Captured images are scanned and landing marker is detected. For this purpose, image features detectors (such as SIFT, SURF or BRISK) are utilized to create a database of keypoints of the landing marker and in a new image keypoints are found using the same feature detector. In this paper results of a framework that allows detection of defined marker for the purpose of drone landing field positioning will be presented.

Key words: Unmanned Aerial Vehicle, Micro Drone, Image Analysis, CCD Camera, Keypoints Descriptors, SIFT, SURF, BRISK, Object Tracking, Camshift

1. INTRODUCTION

This article describes a concept of an autonomous landing system of UAV (Unmanned Aerial Vehicle). This type of device is equipped with the functionality of FPV observation (First Person View) and radio broadcasting video or image data. Additionally often these flying objects are supplied also with advanced on-board equipment capable of performing autonomous flight. The problem is performance of a system of autonomous drone landing in an area with dimensions of $1\text{m} \times 1\text{m}$. Usage of GPS devices for this purpose is not a good option due to their limited accuracy. Therefore to bring down a flying apparatus in a predetermined point then so-called manual landing procedure has to be used, in this scenario a pilot brings the machine on the ground using radio equipment (radio control device). The proposed autonomous landing system based on CCD camera coupled with an image transmission system connected to a base station and a PC determines the direction of movement of the object during the landing procedure (see Fig. 1). It is worth noting that the developed system supports other on-board equipment such as a GPS receiver, barometric sensor, accelerometer sensors, magnetometer and gyroscope.

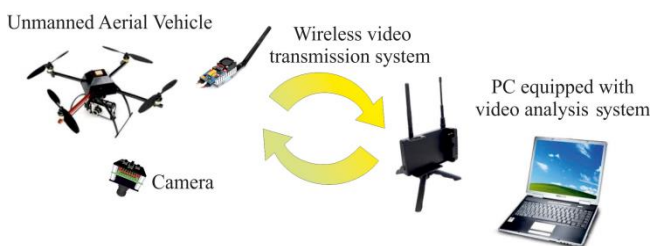


Fig. 1. The general concept of components needed to implement an autonomous landing

2. REQUIREMENTS

In this section main purposes of the presented approach, as well as differences in existing applications will be delineated.

The main purpose of the solution is a possibility of proper detection of landing markers by flying drone in real-time. Markers, representing landing positions, are defined before capturing images on-site and are reused in following runs of the algorithm. The drone captures surroundings using its mobile camera, and as a result of the procedure, the algorithm should give to the drone electronics information about positions of detected landing markers. In overall the algorithm has to be robust with low computational complexity, but also with high accuracy. It's important to note, that the landing markers will be visible in drone's surroundings in different scales and rotations. The solution should handle this situation gracefully, giving to the drone possibility of different capture angles and light intensities, possibly without complicated calibration. Also, captured image can be garbled by radio noise and very often that noise overlaps the landing marker, thus tricking the detection.

To solve the problem above, for the purpose of proper detection of the landing marker even in low quality images, image features detectors are utilized to create a database of keypoints of the landing marker.

Different keypoint extracting algorithms were invented by authors over last many years, these include for example Scale-Invariant Feature Transform (SIFT (Lowe et al., 2004)) or Speeded Up Robust Features (SURF (Bay et al., 2008)), but also recently presented Binary Robust Invariant Scalable Keypoints (BRISK (Leutenegger et al., 2011)) and many others.

3. RELATED WORK

Recognition of defined objects in captured images is a well-known problem and many solutions are existing. Different ap-

proaches to image recognition problems, using texture features classifiers (Skoczylas et al., 2011), image processing filters, etc do exist, and these solutions can perform correct image detection and recognition in captured images as static (Ding et al., 2012), but also moving images (Pan et al., 2013).

Topic of images recognition using keypoints descriptors became recently very popular. For example, SURF algorithm was successfully utilized for face recognition (Li et al., 2013) and road obstacle recognition (Besbes et al., 2010). The BRISK algorithm was implemented into traffic sign recognition as presented in Zheng et al. (2012). These objects are recognized using keypoints detector and descriptors of the points from training images are used to find similar descriptors in captured images. One of such examples is also implementation of SIFT object recognizer in FPGA presented recently (Wang et al., 2013). Authors implemented their algorithm using some of the concepts that are also a base for this publication.

There exist also different approaches to markers recognition as for example recently presented in Yu et al. (2013). Authors are classifying vehicle logos using SVM based on a Bag-of-Words (BoW) approach. The algorithm extracts SIFT features and quantizes features into visual words by 'soft-assignment'. The database is then used for new objects recognition.

Recently, also topic of autonomous detection of markers for Unmanned Aerial Vehicles was researched, for example vision based victim detection system was implemented in Andriluka et al. (2010) where authors successfully used histogram of oriented gradients (HOG) detector for human detection (Dalal et al., 2005). In Shaker et al. (2010) authors presented vision-based autonomous landing system using reinforcement learning.

4. FEATURE DESCRIPTORS

Image feature descriptors are becoming a standard in current state of the art of image recognition algorithms. Their application is mainly for detection and recognition purposes, however there are additional tasks such as medical image registration (Lukashovich et al., 2011).

For this study, author selected most common and popular feature detectors: SIFT, SURF, and BRISK. Main principles of these three algorithms are delineated in the following paragraph.

- Scale-Invariant Feature Transform (SIFT). The SIFT algorithm is quite computational extensive algorithm that detects features in images. Best thing about SIFT is that it is invariant to scale changes, but also rotations and light intensity changes. The original image is incrementally scaled down by a half. This operation generates multiple scale pyramids. Local extrema are identified and key point candidates are detected from which relative orientation is found and removed giving rotation invariance. Orientation histograms over 4x4 regions are extracted and a descriptor vector is created by concatenating all orientation histogram entries. Finally that vector is normalized and a SIFT key point descriptor vector is obtained that contains 128 values.
- Speeded Up Robust Features (SURF). SURF algorithm is also known as an approximate version of SIFT. Main idea of the algorithm is similar, however in SURF authors drew attention to the performance and applied algorithm optimizations. As presented by authors, the algorithm outperforms SIFT in terms of the quality and performance. Scale pyramid

is not constructed as in SIFT, instead different filter sizes (octaves) are used to achieve the same purpose (the scale space is analysed by up-scaling the filter size rather than iteratively reducing the image size [Oyallon et al., 2013]). SURF key point descriptor vector contains 64 values.

- Binary Robust Invariant Scalable Keypoints (BRISK). BRISK is a novel algorithm presented recently. As authors state the quality of key point detection is compared to top state-of-the-art key detector SURF, but needs less computation time. This is achieved by detecting key points in octave layers of image scale pyramid, but also in layers in-between in continuous domain via quadratic function fitting. As a result a bit-vector is constructed.

5. DETECTION OF LANDING MARKERS

First, landing marker images database is created. From all images from the landing markers database keypoints are detected using one of features detector described in previous section. Keypoints feature descriptors associated to landing marker image from the database are stored, these are reference vectors of descriptors used for further detection procedure. Thus, for each marker image, a set D_m of vectors representing features descriptors is created. Note, that these vectors have equal sizes in set D_m , but $|D_m|$ can differ, depending on complexity of the image.

In a new image keypoints are found using the same feature detector, these keypoints form a set $K_c = p_1, p_2, \dots$ and are considered as candidates for keypoints that correspond to the marker image. For all keypoints in the set K_c feature descriptors D_c are calculated, so that each element from set K_c corresponds to one descriptor from set D_c . From the set of candidates K_c , only keypoints that match keypoints existing in the marker image are further processed. For this purpose all keypoints descriptors from training marker D_m are compared with all keypoints descriptors detected in the captured image D_c . A similarity factor, distance (for SIFT and SURF algorithms the similarity factor is distance of vectors, and for BRISK this factor is calculated based on bit operators (Leutenegger et al., 2011)) s of all these descriptors is calculated and keypoints are associated to each other.

Found pairs are filtered to find good matches using technique described in Bradski et al., (2000): first, the minimum distance (min) is found from all matches, and then all distances that are bigger than (this parameter was set empirically; in the literature different parameters for this distance can be found) $2 \cdot min$ are discarded. If calculated distance between two keypoint descriptors is less than a predefined factor, then such keypoint is marked as valid — thus, it is detected and exists in that captured image and is added to the set of found keypoints K_{valid} . Process is repeated for all keypoints from set K_c . If the number of keypoints in set K_{valid} (keypoints in the captured image that correspond to keypoints in marker image, thus with sufficient distance) does not exceed predefined factor $f_v \cdot |D_m|$ thus is not satisfactory, that means that marker object is not visible in the captured image and the algorithm is stopped. In other case, when enough valid keypoints are existing in that captured image, then the algorithm searches for largest cluster of keypoints.

Largest cluster is an area of points that are near to each other. Continuously Adaptive Mean Shift (Camshift) algorithm (Bradski et al., 1998) is applied on the set of detected keypoints K_{valid}

to find largest cluster of keypoints K_{object} . The starting point $p_s = (x_s, y_s)$ for the algorithm is selected from detected keypoints set K_{valid} in such a way that $\forall p = (x, y) \in K_{valid}, \sqrt{(x_s - x)^2 + (y_s - y)^2}$ is minimum. Algorithm iteratively moves point p_s towards center of mass of points inside defined neighbourhood. If the center of mass of points in neighbourhood is equal p_s then algorithm is stopped and cluster K_{object} is found.

And finally, keypoints that reside inside that cluster are considered as keypoints from the marker image. If the number of keypoints inside that found cluster exceeds some predefined factor $f_o \cdot |D_m|$ then that cluster is a detected object marker. The algorithm calculates the center p_o of K_{object} and that center is defined as a landing point for the drone.

All algorithm steps are presented in Fig. 2.

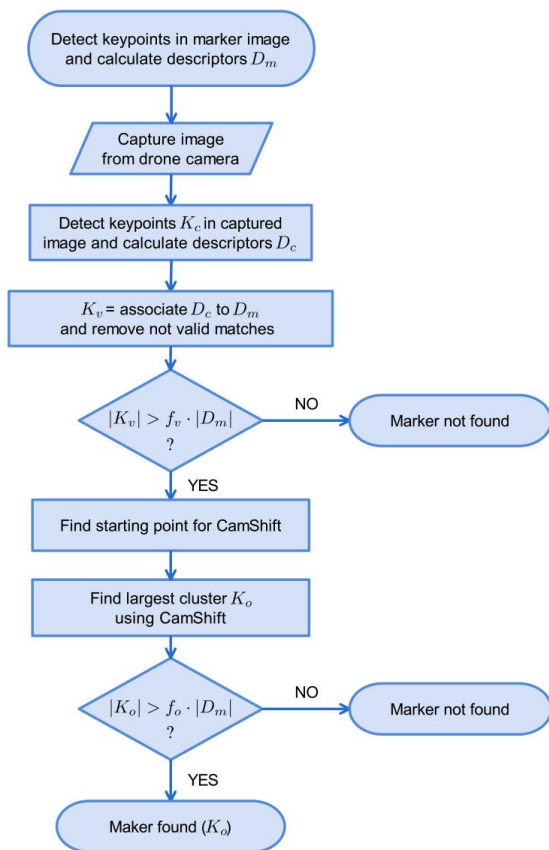


Fig. 2. The Flow chart of marker detection algorithm steps

6. METHODS AND RESULTS

It is very important to select proper keypoint detector algorithm depending on run time and image conditions. Considering, that the whole procedure will be run in an environment with limited resources, the selected algorithm must be robust. Thus, the speed and run time of the algorithm is favored over the accuracy, but from the other hand high accuracy also must be achieved. The keypoint detector is most crucial part for the whole procedure, as it's the most computationally exhaustive part.

To select the keypoint detector algorithm, a database of images was created. Author created a 1m × 1m marker and it was put on the ground in the University campus. Image of the marker is shown in Fig. 3.

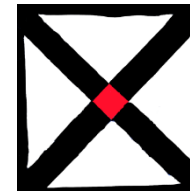


Fig. 3. 1m × 1m marker image used for experiments

Micro drone photographed that marker during the flight above the ground. Videos in real-life scenarios from micro drone camera were captured and several video frames were extracted for further processing. These images presented landing marker in different conditions (rotated, scaled and in different lighting conditions). Such database was evaluated using algorithms described in previous sections.

The first test was performed as follows: on the original learning image a keypoint detector algorithm was run, keypoints descriptors were stored in a resulting set D_{learn} together with their positions K_{learn} on the image. That image was further processed and altered by four tests. Each of the tests was performed independently and all the results of individual steps were combined for better readability.

These test runs included:

- scale was changed to a factor of (0.1,10.0) with a step of 0.1;
- image was rotated and additional black frame was added around the image (rotated by a step of 15° from 0° to 360°);
- a gaussian random noise was applied: $I_{test}(x, y) = I_{orig}(x, y) + random$ from 10 to 1000 with a step of 10;
- lightness was altered: $I_{test}(x, y) = I_{orig}(x, y) + lightness$ from -250 to 250 with a step of 5.

After changes to the image were applied, then the whole detection procedure was performed:

1. On a new image keypoints K were detected.
2. Keypoints pairs were matched and largest cluster was found.
3. Number $|K_{fit}|$ of valid keypoints from the marker image inside that largest cluster in captured image was calculated.
4. To calculate the *ratio*, a number of keypoints $|K_{fit}|$ is divided by a number of all keypoints $|K|$ from the marker image, and multiplied by 100 for readability.
5. *time* to perform calculations was recorded.

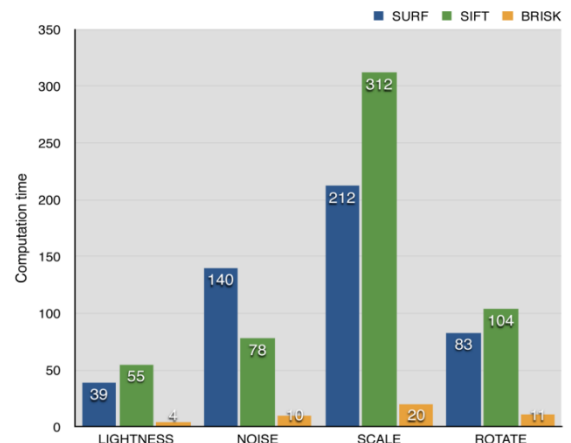


Fig. 4. Comparison of keypoint descriptors computation time in ms for lightness, noise, scale and rotate changes

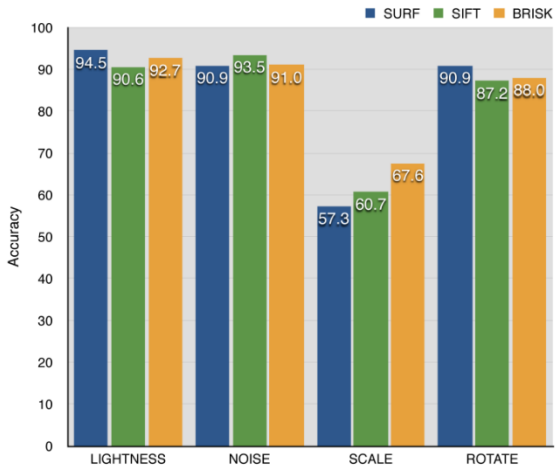


Fig. 5. Comparison of keypoint descriptors accuracy for lightness, noise, scale and rotate changes

On each image from original learning data set a test procedure was run, result *ratios* and *times* needed to perform the test were recorded. From all these tests overall results were created by combining all results into two charts: time and accuracy of each algorithm for lightness, noise, scale and rotate changes. Algorithm was run on 2.6 GHz Intel Core i7 machine. Results of the above computations are shown in Fig. 4 and 5.

During the second approach experimental setup was tested in real-life conditions (see Fig. 6).



Fig. 6. Experimental setup overview

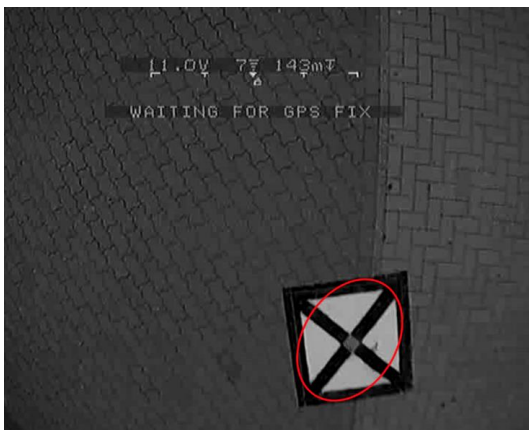


Fig. 7. Example marker detection results in real-life captured image



Fig. 8. Example marker detection result in real-life captured image. Captured image contains radio noise that overlaps the marker

Images were transmitted to a base station and captured. Detection result of the marker in real-life captured image examples are shown in Fig. 7 and Fig. 8.

7. DISCUSSION

Complexity of the whole recognition procedure mainly depends on the number of detected keypoints candidates K_c in the captured image. Also, number of keypoints from original image marker K_m is crucial, as descriptors of both keypoints sets are compared to each other. If the image is complex, the more keypoints are detected and large numbers of keypoints cause a reduction of recognition speed, due to the fact that all these points have to be compared with all keypoints from the marker image.

Recognition accuracy depends on the selected keypoints algorithm, but it is important to note that the algorithm presented in this paper has two factor values that have to be defined empirically. These factors f_v and f_o determine interval when not enough points (or correct points) are found, thus denoting that the marker image is missing in the image (in such case algorithm is stopped). Wrong values of these factors can cause a situation that when marker image is not existing in the image it is in fact incorrectly detected. From the other hand when image has a huge noise, then appropriate settings can guide the algorithm to properly detect even not-readable image, as it is shown in Fig. 9.

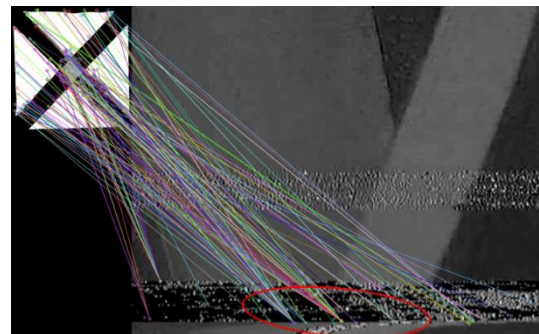


Fig. 9. Example marker detection result in real-life captured image with radio noise, lines show marker keypoints (left-up part of the figure) corresponding to keypoints detected in the captured image (right part of the figure)

Let's analyze result from Fig. 9 that shows correctly detected landing marker when f_o and f_v values were set to low. Due to noise disturbances keypoints are garbled with respect to marker image axes, but the number of detected keypoints is enough to correctly find that cluster. However, that could lead to a situation that when marker is not existing in the captured image it is incorrectly detected as in Fig. 10. Keeping these factors high can cause a situation that marker keypoints is not detected at all and thus false negative results, but too low values cause wrong false positives. Further study is needed to correctly define impact of these factors on recognition procedure accuracy and computation times in real-life scenarios.

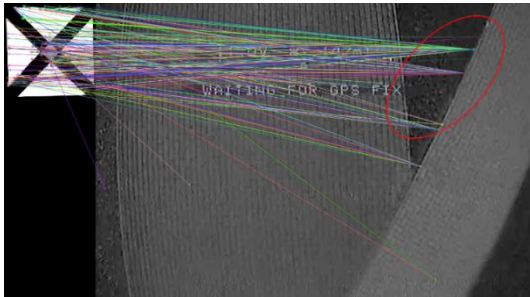


Fig. 10. Example non-existing marker detection result in real-life captured image with incorrectly set f_o and f_v parameters

It is important to note, that from all keypoint detectors, the BRISK algorithm gives the best accuracy for different scales, keeping very good accuracy for rotations. Considering that drone will mostly have to deal with scaled and rotated landing marker image but not noise and lightness changes, this seems as a good feature of this algorithm.

In addition, BRISK outperforms SURF and SIFT in terms of performance considerably, thus BRISK with no doubt is the best choice for further analysis and implementation in embedded environment.

8. CONCLUSIONS

The problem of proper detection of landing marker in user surroundings of the drone can be solved by the algorithm from previous sections. Shapes, representing landing marker, are detected using SURF or BRISK algorithm with very good rate, their low computation time allows the user to run it in limited environment, even in real-time. Proper and fast keypoints matching solves different angles and light intensities, without any calibration, also with existing radio noise. However, it is still necessary to perform further study on the performance of these algorithms in real-life scenarios.

REFERENCES

1. **Andriluka M., Schnitzspan P., Meyer J., Kohlbrecher S., Petersen K., Stryk O., Roth S., Schiele B.** (2010), Vision based victim detection from unmanned aerial vehicles, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.
2. **Bay H., Ess A., Tuytelaars T., Van Gool L.** (2008), Speeded-Up Robust Features (SURF), *Comput. Vis. Image Underst.*, Vol. 110(3), 346–359.
3. **Besbes B., Apatean A., Rogozan A., Bensrhair A.** (2010), Combining SURF-based local and global features for road obstacle recognition in far infrared images. *2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 1869 – 1874.
4. **Bradski G.** (2000), The OpenCV Library, *Dr. Dobb's Journal of Software Tools*.
5. **Bradski R.** (1998), Computer vision face tracking for use in a perceptual user interface, *Proceedings of the Fourth IEEE Workshop on Applications of Computer Vision*, 214-219.
6. **Dalal N., Triggs B.** (2005), Histograms of oriented gradients for human detection, *Computer Vision and Pattern Recognition, 2005. CVPR 2005, IEEE Computer Society Conference*, Vol. 1, 886–893.
7. **Ding D., Yoon J., Lee C.** (2012), Traffic sign detection and identification using SURF algorithm and GPGPU, *SoC Design Conference (ISOCC)*, 506–508.
8. **Leutenegger S., Chli M., Siegwart R.** (2011), BRISK: Binary Robust invariant scalable keypoints, *Computer Vision, IEEE International Conference*, 2548–2555.
9. **Li H., Xu T., Li J., Zhang L.** (2013), Face recognition based on improved SURF, *Third International Conference on Intelligent System Design and Engineering Applications (ISDEA)*, 755 – 758.
10. **Lowé D.** (2004), Distinctive Image Features from Scale-Invariant Keypoints, *Int. J. Comput. Vision*, Vol. 60(2), 91–110.
11. **Lukashovich P., Zalesky B., Ablameyko S.** (2011), Medical image registration based on surf detector, *Pattern Recognit. Image Anal.*, Vol. 21(3), 519–521.
12. **Oyallon E., Rabin J.** (2013), An analysis and implementation of the SURF method, and its comparison to SIFT, *Image Processing On Line*, 1-31.
13. **Pan J., Chen W., Peng W.** (2013), A new moving objects detection method based on improved SURF algorithm, *25th Chinese Control and Decision Conference (CCDC)*, 901–906.
14. **Shaker M., Smith M. N. R., Shigang Y., Duckett T.** (2010), Vision-based landing of a simulated unmanned aerial vehicle with fast reinforcement learning, *International Conference on Emerging Security Technologies (EST)*, 183–188.
15. **Skoczylas M., Rakowski W., Cherubini R., Gerardi S.** (2011), Unstained viable cell recognition in phase-contrast microscopy, *Opto-Electronics Review*, Vol. 19(3), 307–319.
16. **Wang Z., Xiao H., He W., Wen F., Yuan K.** (2013), Real-time SIFT-based object recognition system, *IEEE International Conference on Mechatronics and Automation (ICMA)*, 1361 – 1366.
17. **Yu S., Zheng S., Yang H., Liang L.** (2013), Vehicle logo recognition based on bag-of-words, *10th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 353 – 358.
18. **Zheng Z., Zhang H., Wang B., Gao Z.** (2012), Robust traffic sign recognition and tracking for advanced driver assistance systems, *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 704 – 709.

The work has been accomplished under the funding S/WI/1/2013 financed by Faculty of Computer Science, Bialystok University of Technology.