Obstacle Avoidance for a Quadrotor

Nikoli Dryden * Bryan Plummer †

Abstract

We implement and evaluate the approach of [Lee et al. 2011] for obstacle avoidance for an ARDrone 2 quadrotor, which integrates MOPS and SIFT features to get a sparse representation of obstacles. We also evaluate the existing PTAM [Klein and Murray 2007] approach to UAV navigation. We find that the MOPS features are not robust on the quadrotor's data and that the Lee at al. approach is both ineffective and computationally intensive whereas the PTAM system achieves robust, real-time performance.

1 Introduction

Micro aerial vehicles (MAV's), which include quadrotor and unmanned aerial vehicles (UAV's), have the potential to become more prominent over the next several years. These aircraft allow safe, easy access to many places which are difficult for humans to go, and have applications reaching into search and rescue, tracking, mapping, and others., In order for the higher level tasks to be handled these vehicles must have a reliable navigation system.

There are GPS solutions for navigation that work out of the box (e.g. the AscTec series of quadrotors), but are limited in use to locations where GPS is available. While much progress has been made towards creating a navigation solution for these cases, we still strive to have a system that works in unknown, GPS denied, and cluttered environments. As many MAV's are limited in both carrying capacity and power, laser range finders are too heavy and consume too much power to be of use. Stereo cameras become highly inaccurate after a set distance, rendering them of no more use than using a single camera.

The common structure from motion approach [Dellaert et al. 2000] to visual navigation using a single camera has been shown to suffer from some drawbacks. In order to generate a 3D map, two types of camera translation may be necessary [Shah et al. 2010]. While a quadrotor may be capable of such a maneuver, a UAV is not. In [Shah and Johnson 2009] the amount of computation required using this approach increased by about 15 times with only an increase from 8 to 35 feature points, and the number of features in a real scene can number in the hundreds or thousands.

As an attempt to solve this problem [Lee et al. 2011] proposed an approach to reduce the number of points required to represent the 3D structure of a scene. The authors used multiscale oriented patches (MOPS) [Brown et al. 2005] to create outlines of objects. Since the outlines themselves are unable to tell if an outline is empty or not, they used the 3D location of SIFT features [Lowe 2004] located within the outlines to obtain this information.

Although the proposed method cited collision avoidance for a quadrotor as its intended application, the authors did not dispense any data on its effectiveness in real scenes from images taken using a quadrotor. This paper provides just such an evaluation using images taken from an ARDrone 2 quadrotor with the processing being performed on a laptop obtaining the images from the quadrotor through a wireless network. In addition, we evaluate the use of the parallel tracking and mapping (PTAM) [Klein and Murray 2007] approach that was adapted as an online solution for quadrotors in [Weiss et al. 2011].

*dryden2@illinois.edu

†bplumme2@illinois.edu

2 Related Work

Due to weight and energy restrictions aboard MAV's, a vision based approach is seen as one of the most viable navigation solutions in GPS denied environments. Progress in using visual senors for navigation purposes include performing specific tasks such as in [Johnson et al. 2005] where a single camera was used to guide and land a MAV, or [Huang et al. 2011] where an RGB-D camera was used to map an environment and localize a quadrotor within that map, but is restricted for use in indoor environments.

Another common approach in the literature uses optical flow to estimate the relative motion between the frames of a camera. In these systems one is able to detect collisions by measuring the relative rate of expansion of objects. This method behaves poorly when light intensity does not remain constant [Horn and Schunck 1981] and tends to be sensitive to noise and an unstable camera. As all MAV's will contribute to some vibrations in the camera this leads to inaccurate estimates in flow vectors.

Some recent work attempts to use image segmentation to identify an obstacle and build a dense map around it [Ha and Sattigeri 2012]. This approach shares some motivations as the first method attempted in this work in that it attempts to reduce the computation required for navigation by using a limited number of feature points. As with [Lee et al. 2011], this approach has only been evaluated on theoretical data. The authors note that in some cases image segmentation is not possible and intend to evaluate and extend their work to ascertain the extent this kind of information is beneficial to navigation.

An approach specifically tailored for low-cost quadrotors is described in [Engel et al. 2012b; Engel et al. 2012a]. The system they develop is designed for use with quadrotors paired with a laptop for processing, and builds upon the PTAM monocular SLAM system. The authors have demonstrated their approach with ARDrone quadrotors, and we make use of their system to evaluate and compare PTAM with [Lee et al. 2011].

3 Approach

We begin by describing our implementation of the approach in [Lee et al. 2011]. Subsequently, we introduce a variation we attempted to generate an improved 3D map than we obtained using the unmodified version of [Lee et al. 2011]. Lastly, we describe our integration of the PTAM system with the quadrotor.

3.1 Lee et al.'s Method

Our approach closely follows that in [Lee et al. 2011]. The goal is to make use of MOPS features to limit the number of SIFT features necessary for computing structural information about obstacles. We begin by assuming that we have, as input, two images taken from the quadrotor's forward-facing camera, that the images are of the same scene, and contain corresponding features. For our purposes, we need only a greyscale image and so discard the colour information.

In each image, we compute MOPS descriptors [Brown et al. 2005] (see Figure 3). The descriptors are located at features detected by a multi-scale Harris corner detector. Each descriptor is an 8×8 patch of bias/gain normalized intensity values and is oriented using

a blurred local gradient. Feature density is controlled using an adaptive non-maximal suppression algorithm. SIFT descriptors [Lowe 2004] are also computed (see Figure 2), using the OpenCV computer vision library [ope].

Descriptors in these two images are then matched in order to generate putative matches (see Figures 2 and 3). For MOPS, matching is performed using a brute-force matcher, since existing k-d tree implementations in OpenCV do not support efficient dynamic modification after creation. SIFT features are matched using OpenCV's FLANN fast approximate nearest-neighbour [Muja and Lowe 2009] matcher.

These matches are then used to determine the location of the detected feature points in three-dimensional space via triangulation. Our approach to triangulation is based on the one implemented in PTAM [pta]. A homography is first computed between matching points. This homography is then decomposed into rotation and translation matrices [Malis and Vargas 2007] which are used to determine a projection matrix. This is used to triangulate points as usual.

In our implementation, the above steps involving SIFT and MOPS are implemented in parallel using the standard pthreads library. Since to this point there are no interactions between the SIFT and MOPS features and the input images are constant, the operations are independent. We thus run each in a separate thread in order to maximize performance.

The locations of MOPS features indicate the corners of objects in the world. The orientation of the descriptors is used to extract the outlines of objects to obtain edge spatial information. However, this does not provide information on the internal structure of the object; we use the SIFT features to achieve this. We consider SIFT features within the outlines constructed by MOPS features, which provide information on the internal structure of the objects: the distance to the points will indicate the type and extent of the object. An example of merged SIFT and MOPS features from [Lee et al. 2011] is provided in Figure 1.

We now determine the "type" of the object, based on comparison of the distance of SIFT features within an outline to the distance to MOPS features on the outline. Define D_M to be the distance to a MOPS feature, and D_S to be the distance to a SIFT feature within the outline. Let T_1 and T_2 be two threshold values. We have the following equations

$$|D_M - D_S| \leq T_1 \tag{1}$$

$$T_1 < |D_M - D_S| \leq T_2 \tag{2}$$

$$T_2 < |D_M - D_S| \tag{3}$$

T_1 should be set such that there is a high probabilitiy that the SIFT and MOPS features are on the same object, and T_2 such that there is a high probability that the features are on different objects. We thus have the following cases:

- Equation 1 is the case of an object with a closed interior.
- Equation 2 is again the case where an object has a closed interior, but possibly with a convex shape.
- Equation 3 is the case of an object with an open interior.

In the third case, we repeat the above process feature points on an object behind the open space.

This comparison allows us to determine what sort of obstacle an object is and to locate where in the world the objects exist, relative to the quadrotor.

3.2 Replacing MOPS with Contours

After finding object outlines using MOPS features to not give sufficient information, we sought an alternative approach. As [Suzuki and Abe 1985] has been widely tested and locates contours within an image it appeared to be sufficient to meet our needs. We used the Canny Edge Detector [Canny 1986] to convert to binary images as the input to this method. After locating the contours in an image we marked the key points found using the SIFT algorithm as belonging to an object outline and proceeded as normal through the rest of the approach of [Lee et al. 2011] (see Figure 4).

3.3 Integrating PTAM

Integrating PTAM with the quadrotor is remarkably simple. We made use of two different methods to accomplish this, in order to have different features available. Both methods rely upon the Robot Operating System (ROS) [ros] for support and the ardrone_autonomy [ard] driver for communication with the quadrotor.

Our first approach is to directly test PTAM on data from the quadrotor. For this, we made use of the ethzasl_ptam package [eth], which integrates PTAM with ROS. It was then only a matter of converting the camera stream from the ardrone_autonomy driver to the requisite format for use with PTAM and calibrating the camera appropriately.

Secondly, we made use of the tum_ardrone [tum] package for ROS, which integrates a version of PTAM with an extended Kalman filter for data fusion and state estimation and a system for generating steering commands for the quadrotor. tum_ardrone builds upon PTAM to provide an environment specifically designed for use with the ARDrone and ARDrone 2, and in addition to obstacle detection, provides a system for navigation and auto-pilot.

4 Experiments

Our data sets consist of samples obtained by recording output from the quadrotor's forward-facing camera. These are 640×360 pixel colour images saved in JPEG format recorded at a rate of one per each 0.01 second. Our initial data sets consisted of

- A set of 212 frames in which the drone is sitting still on a table.
- A set of 321 frames in which the drone takes off, flies down a Siebel Center hall, and lands.
- A set of 301 frames in which the drone takes off, flies through the Siebel Center atrium, and lands.

These were supplemented with the following additional data sets from the quadrotor, this time recorded at a rate of one per each 0.1 second.

- A set of 130 frames in which the drone flies through the Siebel Center atrium with additional movement, and people present.
- A set of 275 frames in which the drone flies closely along the pillars and display cases of in the Siebel Center atrium.
- A set of 49 frames in which the drone flies towards a corner in the Siebel Center atrium.

We additionally conducted experiments in which we evaluated methods on live data from the quadrotor while it was flying.

Figure 1: *Example output for combined SIFT and MOPS features.*

4.1 Lee et al.

We began by evaluating the SIFT and MOPS features on samples from our data sets. As this is the basis upon which the rest of the approach is built, this is a reasonable place to begin. SIFT keypoints and the computed matches for a sample image are shown in Figure 2. As can be seen, the matches between keypoints in general appear to be reasonable.

Figure 3 displays results for MOPS keypoints and matches on the same frame pair as above. These are quite poor, and further discussion is made of this in the analysis section. Additionally, it was found that extraction of MOPS descriptors and matching took an much more time than would be appropriate for this application (several seconds) in any situation other than highly constrained thresholds to eliminate most points.

To validate that our feature point detection for MOPS was correct, we took several sample images from our data sets and ran the VLFeat [vlf] multi-scale Harris corner detector on the images. Results for one such image are displayed in Figure 5. It should be noted that the detected keypoints in that image are quite similar to those detected in the MOPS sample images; as is expected, since MOPS makes use of a multi-scale Harris corner detector.

4.2 Contours

Subsequently we replaced MOPS features with contours on which we detected SIFT keypoints; see Figure 4 for example results, again on the same frame pair. While these results are better than with MOPS, they again are insufficient.

4.3 PTAM

PTAM was evaluated on a live-stream from the quadrotor as it flew, as it could not be meaningfully evaluated on the sample data sets we had recorded. PTAM was able to successfully initialize and maintain a map of its environment which included the relative distance from the quadrotor to objects. See Figure *** REF FIGURE *** for an example of the map produced.

This performance was sufficient for real-time operation, and we were able to successfully fly the quadrotor through environments using the tum_ardrone package.

5 Analysis

During our experiments we considered the following factors as the primary characteristics the algorithm must possess in order to be be appropriate for our task:

1. It must robustly produce an accurate 3D model of the world.
2. It must run in real time.

If an algorithm is unable to create an accurate representation of a scene then it would be unsuitable to use for navigation purposes. On the other hand, if it is unable to run in real time collision information may not be available until it is too late to avoid the obstacle. In doing so, both factors were considered equally important in our evaluation.

As [Lee et al. 2011] proposed to use a sparse representation in the pursuit of real time performance, one of our primary concerns was if the 3D model it creates would be suitable for our task. Upon inspecting the results of the tests described in Section 4.1, it was obvious that the matches were very noisy, and the subsequent reconstruction using the punitive matches did not produce a model that was representative of the scene. Of particular note in the case of the example shown was the few number of features detected along the pillar. The lack of many MOPS features on the pillar indicates that our object outlines would not be very representative of its true location, making autonomous navigation risky. While this by itself

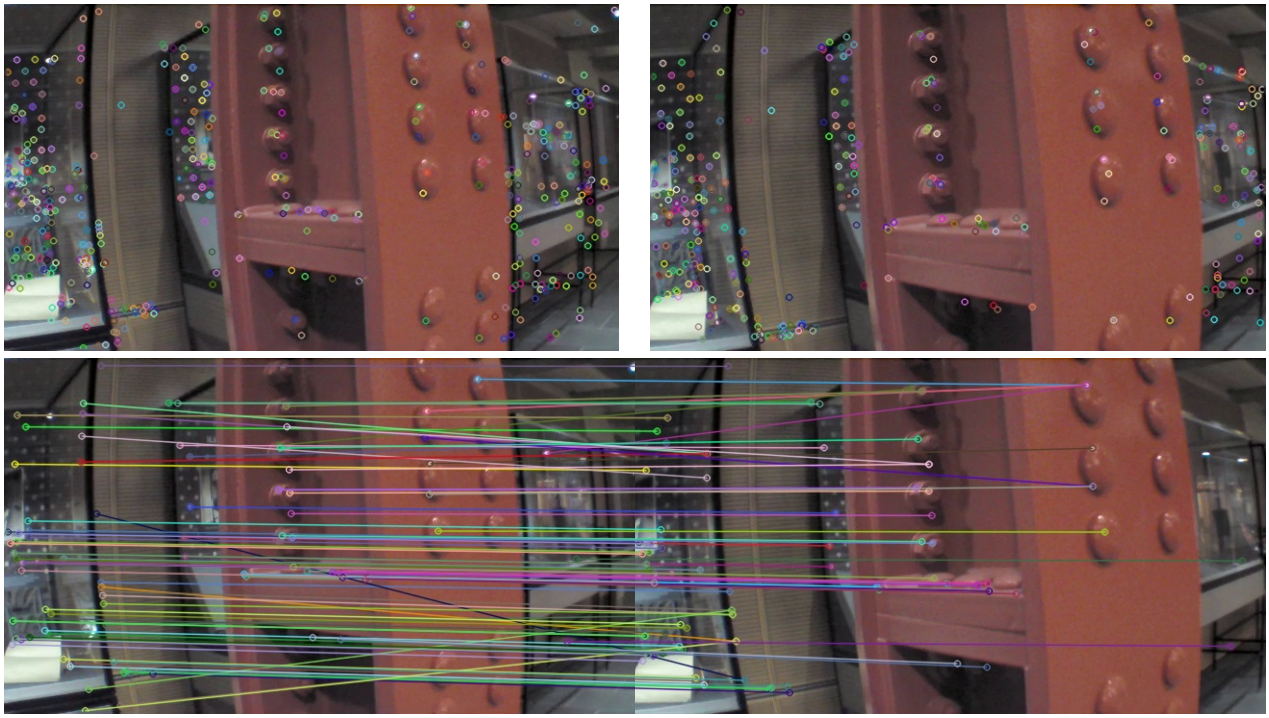

Figure 2: *SIFT keypoints and matches.*

intensified the misgivings on the approach, it was not clear that an alternative to the MOPS features could not produce a better representation as the SIFT features did provide better punative matches.

This was reinforced with our test using image contours rather than MOPS features as we were able to create a better set of punative matches between the images. While the reconstruction using these matches still produced a 3D model that would not be sufficient for our task, it does indicate that additional work could be performed in this space that would see better results. In addition, it is important to note that while the display cases in the presented example present additional challenges for any reconstruction effort, the algorithm did not produce good results in images without any glass in the frame.

Even though some progress was made to create a more accurate representation, it came at the cost of increasing the number of features in a scene and the computational effort that would be necessary for navigation. Since we focus on reducing the number of keypoints used for matching and reconstruction, it also amplifies the effect poor matches have on our algorithm making it sensitive to noise. From this analysis it follows that if we were able to compute a large number of features very quickly, even if we had several bad matches they would have less effect on our solution.

Using this line of reasoning we explored the effectiveness of the PTAM approch. This algorithm has been demonstrated to perform at 20Hz on a computationally limited platform (an ATOM 1.6GHz single core processor) using up to 300 keypoints in each image to create the punative matches with a RMS error in its position of 3.02cm or less along any axis [Weiss et al. 2011]. This approach is not without its own drawbacks, however.

In [Weiss et al. 2011] they recommend at least 90 percent overlap between frames for good tracking. In our own test flights using this system we found that the quadrotor's position was lost in the map when stopping suddenly, and rotations also contributed to errors and should be avoided as much as possible. As with many other vision based applications, textureless surfaces continue to be a problem and can result in loss of a quadrotor's position. This seems to point to using a downward facing camera to avoid rotations and help increase the probability of a feature rich map by using a wide field of view camera.

6 Individual Contributions

Individual contributions break down roughly as follows. Nikoli was responsible for:

- Implementing SIFT feature detection, extraction, and image matching
- Implementing 3-D reconstruction and triangulation
- Implementing the multi-threaded image pipeline
- Setting up and integrating ethzasl_ptam with the quadrotor
- Setting up tum_ardrone
- Experimental evaluations
- Presentation and paper writing

Bryan was responsible for:

- Implementing MOPS feature detection, extraction, and image matching
- Experimenting with a contour-based technique to replace MOPS
- Assistance with 3-D reconstruction
- Assistance with setting up ethzasl_ptam
- Experimental evaluations

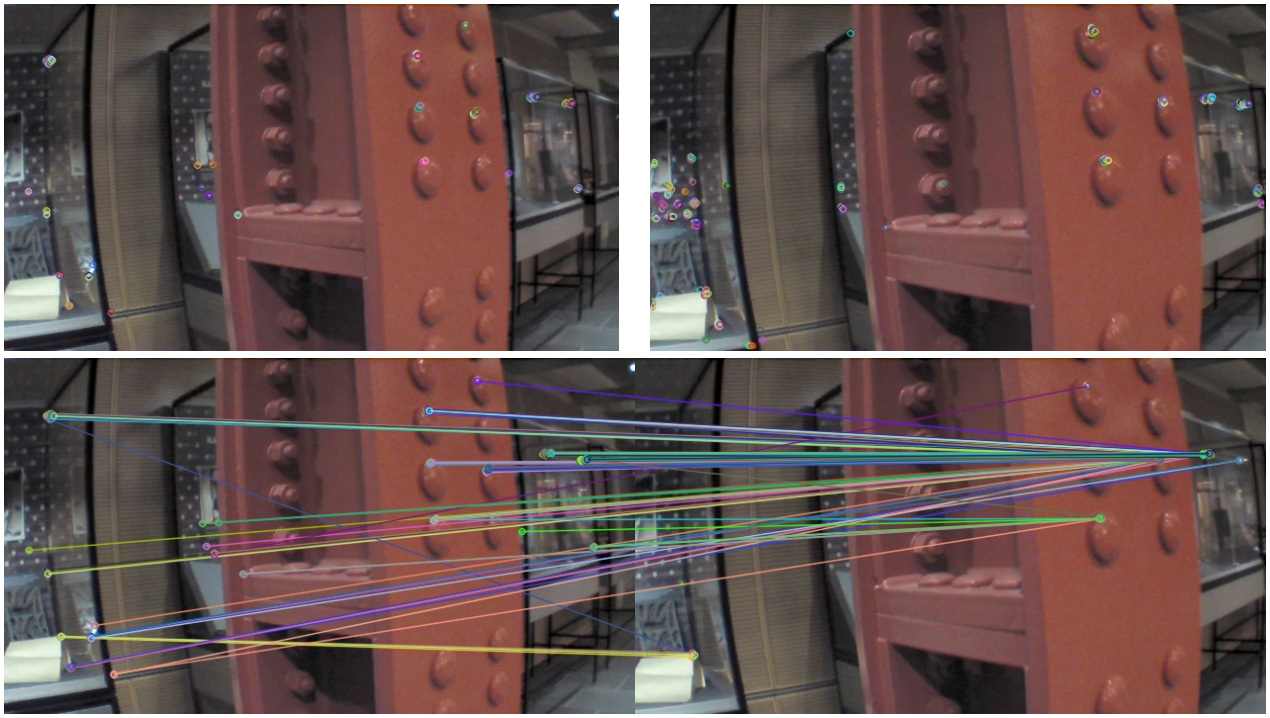

Figure 3: *MOPS keypoints and matches.*

- Presentation and paper writing

Because MOPS was implemented from scratch, we feel that the work was split quite evenly.

References

ardrone_autonomy. `https://github.com/AutonomyLab/ardrone_autonomy`. Accessed: 2013-05-06.

BROWN, M., SZELISKI, R., AND WINDER, S. 2005. Multi-image matching using multi-scale oriented patches. In *CVPR*.

CANNY, J. 1986. Computational approach to edge detection. *PAMI 8*, 6, 679–698.

DELLAERT, F., SEITZ, S., THORPE, C., AND THRUN, S. 2000. Structure from motion without correspondence. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, 557–564 vol.2.

ENGEL, J., STURM, J., AND CREMERS, D. 2012. Accurate figure flying with a quadrocopter using onboard visual and intertial sensing. vol. 320, 240.

ENGEL, J., STURM, J., AND CREMERS, D. 2012. Camera-based navigation of a low-cost quadrocopter. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2815–2821.

ethzasl_ptam. `http://ros.org/wiki/ethzasl_ptam/`. Accessed: 2013-05-06.

HA, J., AND SATTIGERI, R. 2012. Vision-based obstacle avoidance based on monocular SLAM and image segmentation for UAVs. In *Proc. AIAA Infotech@Aerospace Conference*.

HORN, B., AND SCHUNCK, B. 1981. Determining optical flow. *Artificial Intelligence 17*, 1–3, 185–203.

HUANG, A. S., BACHRACH, A., HENRY, P., KRAININ, M., MATURANA, D., FOX, D., AND ROY, N. 2011. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *Int. Symposium on Robotics Research (ISRR)*.

JOHNSON, A., MONTGOMERY, J., AND MATTHIES, L. 2005. Vision guided landing of an autonomous helicopter in hazardous terrain. In *IEEE International Conference on Robotics and Automation*.

KLEIN, G., AND MURRAY, D. 2007. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*.

LEE, J., LEE, K., PARK, S., IM, S., AND PARK, J. 2011. Obstacle avoidance for small UAVs using monocular vision. *Aircraft Engineering and Aerospace Technology 83*, 6, 397–406.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *IJCV 60*, 2, 91–110.

MALIS, E., AND VARGAS, M. 2007. Deeper understanding of the homography decomposition for vision-based control. Tech. rep.

MUJA, M., AND LOWE, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09*, INSTICC Press, 331–340.

OpenCV. `http://opencv.org/`. Accessed: 2013-05-06.

Parallel tracking and mapping for small ar workspaces. `http://www.robots.ox.ac.uk/~gk/PTAM/`. Accessed: 2013-05-06.

Robot Operating System. `http://www.ros.org/`. Accessed: 2013-05-06.

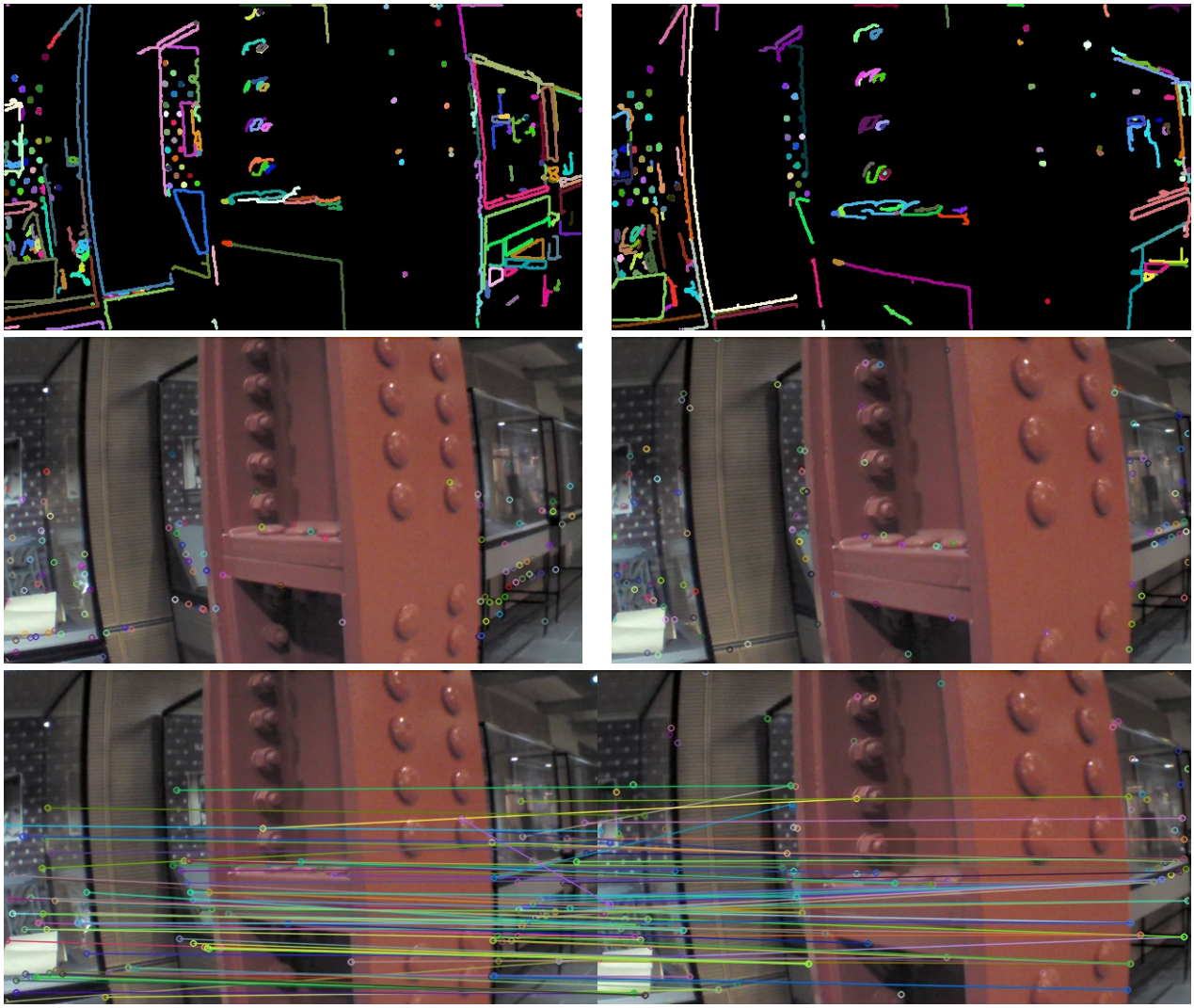

Figure 4: *Contour lines, keypoints, and matches.*

SHAH, S. I. A., AND JOHNSON, E. N. 2009. 3D obstacle detection using a single camera. In *Proc. AIAA Guidance, Navigation, and Control Conference*.

SHAH, S. I. A., KANNAN, S., AND JOHNSON, E. N. 2010. Motion estimation for obstacle detection and avoidance using a single camera for UAVs/robots. In *Proc. AIAA Guidance, Navigation, and Control Conference*.

SUZUKI, S., AND ABE, K. 1985. Topological structural analysis of digitized binary images by border following. 32–46.

tum_ardrone. `http://www.ros.org/wiki/tum_ardrone`. Accessed: 2013-05-06.

VLFeat. `http://www.vlfeat.org/`. Accessed: 2013-05-06.

WEISS, S., SCARAMUZZA, D., AND SIEGWART, R. 2011. Monocular–SLAM–based navigation for autonomous micro helicopters in GPS–denied environments. *Journal of Field Robotics 28*, 6, 854–874.

Figure 5: *VLFeat multi-scale Harris corner detector keypoints.*