Measurement 81 (2016) 264–269

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

Embedded, real-time UAV control for improved, image-based 3D scene reconstruction

Jean Liénard [a], Andre Vogs [b,1], Demetrios Gatziolis [c], Nikolay Strigul [a,*]

[a] Department of Mathematics and Statistics, Washington State University Vancouver, Vancouver, WA 98686, USA
[b] Intel Inc., Hillsboro, OR 97124, USA
[c] USDA Forest Service, Pacific Northwest Research Station, Portland, OR 97204, USA

ARTICLE INFO

Article history:
Received 11 August 2015
Received in revised form 3 December 2015
Accepted 14 December 2015
Available online 18 December 2015

Keywords:
3D reconstruction
Aerial robotics
Computer vision
Robotics in agriculture and forestry
Real-time photogrammetry

ABSTRACT

Unmanned Aerial Vehicles (UAVs) are already broadly employed for 3D modeling of large objects such as trees and monuments via photogrammetry. The usual workflow includes two distinct steps: image acquisition with UAV and computationally demanding post-flight image processing. Insufficient feature overlaps across images is a common shortcoming in post-flight image processing resulting in the failure of 3D reconstruction. Here we propose a real-time control system that overcomes this limitation by targeting specific spatial locations for image acquisition thereby providing sufficient feature overlap. We initially benchmark several implementations of the Scale-Invariant Feature Transform (SIFT) feature identification algorithm to determine whether they allow real-time execution on the low-cost processing hardware embedded on the UAV. We then experimentally test our UAV platform in virtual and real-life environments. The presented architecture consistently decreases failures and improves the overall quality of 3D reconstructions.

1. Introduction

The interest in and popularity of Unmanned Aerial Vehicles (UAVs) have been increasing rapidly [12,7]. In particular, UAV-based, photogrammetric 3D reconstruction of large objects has been employed in forestry [10] and archaeology [26]. In these practical applications, UAVs serve simply as platforms carrying low-cost, off-the-shelf digital cameras that acquire images along a preprogrammed, GPS-enabled trajectory at a predetermined frame rate [12]. Dynamic trajectory adjustments enable obstacle avoidance [22], aggressive maneuvers [17], landing [2,18,25], and frontier-based exploration [11]. Image processing can be either outsourced to a remote computer through a WiFi connection [21,22] or performed on-board, as demonstrated by Meier et al. [16] who developed a complete UAV platform with an embedded computer. On-board image processing, a promising direction for photogrammetry applications, is currently heavily constrained by the computational abilities of embedded hardware [5,16]. Real-time processing of imagery acquired by a UAV and based on a standard stereo camera arrangement has been often used for Simultaneous localization and mapping (SLAM, [16]) while other UAV architectures have relied on two or more cameras in various configurations [12,27,31]. For example, Yang et al. [31] combined non-overlapping downward- and forward-oriented cameras to track features across a larger field of view. Shen et al. [27] relied on two cameras with different frame rates and fusion of monocular and stereo approaches to estimate the UAV's position.

In the typical UAV-based photogrammetry workflow, acquired HD photographs (5–12 MP) are processed after

* Corresponding author.
E-mail address: nick.strigul@wsu.edu (N. Strigul).
[1] A. Vogs contributes to this work as a volunteer at Washington State University.

http://dx.doi.org/10.1016/j.measurement.2015.12.014
0263-2241/

the flight, on a computer system equipped with adequate hardware resources [26,10,30]. The independence between the image acquisition and image processing phases is the most serious shortcoming of this workflow because it often leads to scene reconstruction failures attributed to insufficient feature overlap between images [10]. Such failures can be costly, as when the scene is located in a remote area, or irreversible, as where the scene has been altered permanently after the UAV flight (e.g. forest stand burned). A naive solution to avoiding these failures would be to increase the acquisition frame rate while reducing the speed of the UAV platform [10]. However, such measures address only indirectly the core problem of non-overlapping features across images, which also depends on other factors relative to image quality during the flight (such as changing illumination conditions, complexity of dominant scene objects, or movement of objects by wind). In practice, increasing the frame rate reduces image clarity and poses technical challenges relative to the time required to store HD photographs, resulting in a sharp increase in cameras cost. An affordable UAV system allowing consistently complete reconstructions of scene objects must therefore be able to handle changes in image feature density during a flight, a condition that requires simultaneous image acquisition and processing. We address this requirement in the present study and introduce a low-cost solution that detects and compensates for insufficient feature matches between sequentially acquired images by employing on-board, real-time image processing and ultimately guarantees complete 3D reconstruction of the targeted objects.

2. Methods

2.1. UAV platform

We built a custom UAV (Fig. 1) using a DIJ F550 Hexacopter Frame, a 3DR Pixhawk autopilot equipped with GPS and compass (3D Robotics Inc.), a 915 MHz telemetry radio to the ground station computer, a FrSky receiver, a Spectrum DX7 transmitter, a Tarot T-2D Brushless gimbal, and 3 cell LIPO batteries which sustained a 20-min flight. The UAV carries an open-source, single-board Minnowboard Max computer (Intel Inc.) featuring a dual core ATOM CPU at 1.33 GHz, 2 GB of DD3 RAM, a GPU chipset, and a set of input/output ports adequate for platform control and functionality. This board has low power consumption (less than 10 W), which is convenient for UAV applications.

Our UAV platform shares similarities to the one developed by [16], where image frames acquired with low resolution (0.3 MP), high frame rate, low latency, stereo CCD cameras were processed in real-time on an embedded computer. Instead, our platform carries two imaging systems: (1) an inexpensive, low-resolution web camera that delivers a stream for real-time processing and (2) a high-resolution GoPro camera (5–12 MP) triggered through its serial connection. We use a single-board computer (Intel's Minnowboard Max) in lieu of the combination of the custom-built baseboard and Computer-on-Module used by [16]. Our configuration supports expeditious system assembly and has lower cost, but also limited customization options for its base components.

2.2. Real-time keypoint detection

SIFT [14] is the most widely used algorithm for detection of image keypoints; more recent alternatives include SURF [3], ORB [24] and BFROST [9]. SIFT is part of our current postprocessing workflow [10] and while slower than its competitors, it allows more efficient keypoint matching [9,4,6]. We benchmarked several open-source implementations of the SIFT algorithm with the on-board computer including Ezsift,[2] a C++ implementation; the OpenCV implementation[3]; SIFT_PyOCL, an implementation exploiting GPU with OpenCL [20]. To compare these libraries and assess the feasibility of real-time feature detection, we measured the time needed to process the same set of images. We also considered the idealized scenario where pictures have already been loaded into memory, and the on-board computer does not perform trajectory control nor communicates with the autopilot.

2.3. Real-time active control

Prior experience [10] suggests that the number of keypoints matched across images is a reliable indicator for a successful 3D scene reconstruction. Our prior data acquisition approach was acquiring images at specific GPS waypoints along the drone trajectory preprogrammed with PIXHAWK autopilot. The addition of the single board computer enables a feedback control loop in this image acquisition scheme. The overall goal of the active control strategy is thus to ensure that the number of keypoint matches is adequate for the post flight reconstruction of the scene. The implemented active control strategy consists of UAV backtracking to the halfpoint between the last two GPS waypoints to take an additional high-resolution picture, if the number of keypoints matched between two sequential images falls below a predefined threshold.

Specifically, when the UAV arrives at a mandatory waypoint, it starts loitering and the following two events occur simultaneously. The high-resolution camera is triggered to take a picture. At the same time, the single-board computer starts identifying keypoints from the latest low-resolution frame of the web camera. As soon as keypoints are extracted and matched against those at the previous location the UAV moves toward the next waypoint if the matching ratio is high enough, or backtracks to the halfway point if the matching ratio is below the set threshold.

2.4. Virtual reality experiments

We relied on simulation to validate the active control algorithm and troubleshoot our implementation. We first generated a virtual environment and rendered it using the open-source program POVRAY (Persistence of Vision

[2] http://sourceforge.net/p/ezsift/wiki/Home/.
[3] Version 2.4.11, http://opencv.org.

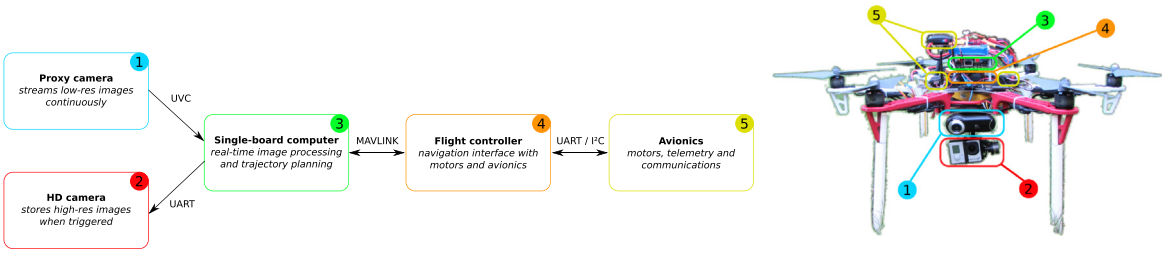

Fig. 1. UAV architecture developed in this work. Boxes depict system components and arrows communication protocols.

Raytracer, povray.org). A virtual reality scene featuring a deciduous tree surrounded by buildings was rendered on the screen of a laptop and subsequently captured by the UAV low-resolution camera oriented towards the screen. Real-time control was enabled via ethernet connection between the embedded computer and the laptop, allowing the transfer of plain-text directions to the virtual flight controller running on the laptop, in lieu of communication with the PIXHAWK used in real flights.

During the virtual flights, the virtual camera was always oriented towards the tree. By design, the simulated environment had two levels of complexity relative to the matching of keypoints (Fig. 2). The half of the trajectory labeled as "feature rich" and shown in green in Fig. 2, was designed to display numerous visual cues. It contained a deciduous tree, the targeted scene object, as foreground and two multistory buildings with regular arrangement of windows as background. These buildings produced a large number of keypoints, matched easily across images. The other half of the trajectory, labeled "feature-poor" and colored pink, had the horizon of a neutral looking scene as background and therefore, only a limited number of cues.

2.5. Field experiments

Communication between the flight controller and the on-board computer was achieved via the MAVLINK protocol (Fig. 1), enabling two navigation modes: "guided", in which the onboard computer is continuously passing instructions to the flight controller, and "manual", when the flight controller is operated remotely by a human. Manual flight control was used only for UAV takeoff and landing, and as a safety precaution in the event of an unforeseen emergency. We used MAVProxy[4] to handle the low-level processes involved in the communication with the flight controller.

2.6. 3D scene reconstruction

To perform 3D reconstruction and thus validate the effectiveness of the active flight control process, we relied on our previously monocular-based workflow [10]. In brief, images from the high-quality camera were first undistorted to remove lens aberration and then processed using VisualSFM [29] to produce a sparse scene model. We then

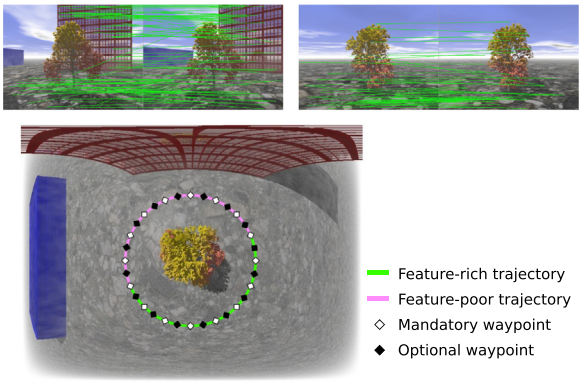

Fig. 2. Virtual reality environment. Top: Two pairs of pictures taken from consecutive waypoints, with green lines indicating matching keypoints. Bottom: Panoramic nadir view, centered on the tree, illustrating the backtracking strategy. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

employed SURE [23] to perform the dense reconstruction. Postprocessing this 3D reconstruction was performed on desktop computers.

When feature overlap is insufficient, the 3D reconstruction workflow results in several disconnected models, each model capturing some part of the object of interest [10]. In this work, the number of 3D sparse models and tally of points in the associated dense reconstruction cloud were used as simple metrics of scene reconstruction completeness and quality.

3. Results

3.1. Keypoint extraction time

Benchmarking of keypoint extraction from 187 high resolution, 5 MP flight pictures on the MinnowBoard Max computer revealed differences in execution time larger than one order of magnitude among the SIFT algorithm implementations (Table 1).

Not surprisingly, the SIFT_PyOCL implementation which exploits the GPU outperformed all CPU implementations, but it still required almost 8 s to process a single high-resolution image, a duration clearly incompatible with real-time processing. Thankfully, the workload can be reduced and efficiency promoted with two image manipulations: cropping centered on the image region of interest, or degradation of image resolution. Lowering the

[4] http://tridge.github.io/MAVProxy/.

Table 1
SIFT running time for one 5 MP picture (in seconds).

Implementation	Mean	Standard deviation
Ezsift	110.6	0.6
OpenCV	22.5	0.3
SIFT_PyOCL	7.6	0.2

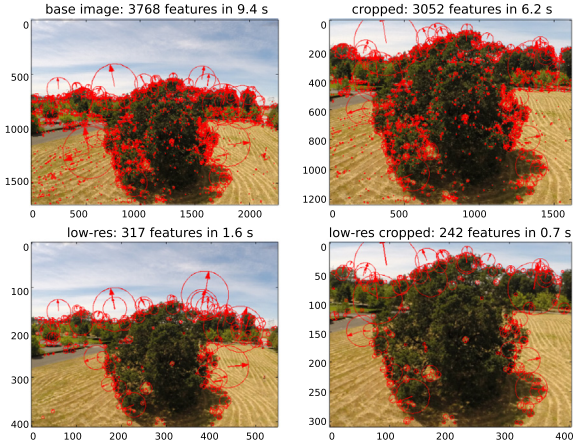

Fig. 3. Strategies for decreasing the number of identified image keypoints and running time, while preserving the most important image features. From top to bottom and left to right: base image, cropped, low-resolution, and cropped low-resolution image. Each keypoint is represented by a circle and an arrow, indicating scale (importance) and direction.

image resolution yielded the largest time gains (Fig. 3) and made real-time processing possible. In the applications presented in the next sections, we rely on a web camera with resolution 640×480 pixels, providing the stream on which real-time keypoint detection is performed.

3.2. Validation in virtual reality environment

The reference trial, without active control, generated $n = 6$ disconnected models (Fig. 4A and B). With the keypoint matching threshold between successive images set to 30%, the active control mechanism enforced backtracking primarily on the feature poor portion of the trajectory and led to a single unified model ($n = 1$, Fig. 4C). The comprehensive 3D model obtained with backtracking had 169,613 points, while the combination of the disconnected models obtained without backtracking summed to only 76,174 points. To confirm that the observed model improvement was not an artifact of using more images,

Fig. 4. Illustration of the quality of 3D reconstructions in the virtual environment. A and B: two partial models produced without backtracking. C: the single model obtained when active control is used. The tree is barely recognizable in the disconnected models (A and B), while the single unified model captures its structure in greater details (C).

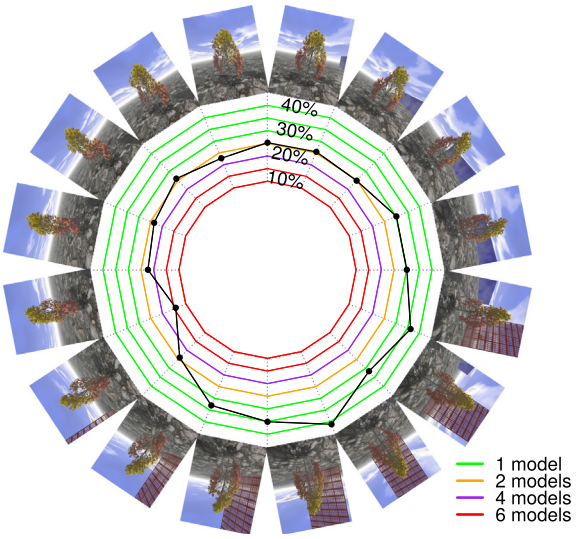

Fig. 5. Variability in keypoint matching ratio between adjacent virtual images generated at mandatory trajectory waypoints (black). Radial isolines denote keypoint matching levels at 5% increments and are colored by the number of 3D scene models reconstructed by processing the virtual image set. Graph orientation adopted from Fig. 2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

we also conducted 10 additional virtual flights in which the UAV was instructed to backtrack at random, unrelated to the number of keypoints matched between adjacent images. It was thus corroborated that in the absence of backtracking, the number of reconstruction failures was higher, with mean $\bar{n} = 2.5$ models (± 0.27, two sided t-test significant with $p < 0.05$).

The systematic study of keypoint matching ratios between adjacent images confirmed that the parts of the trajectory from which buildings were not visible had indeed lower matching ratios (Fig. 5). Increasing the matching threshold from 10% (no backtracking) up to 45% (UAV backtracking at every waypoint) at 5% increments, resulted in progressively fewer disconnected models. Thresholds at or above 30% lead to a single unified model.

3.3. Validation in real-life environment

An example of a complete reconstruction obtained by using the same process as in simulation but applied to images acquired during an actual UAV flight is shown in Fig. 6. In this case, the keypoint matching threshold used to trigger platform backtracking was set to 60%. It is higher than the 30% used with the virtual reality imagery, owing to differences in overall scene complexity. The 60% value was determined empirically, according to experience from numerous prior experiments with actual UAV imagery, where a minimum 55% keypoint matching rate between adjacent images was found to be necessary for single-model reconstruction. Qualitatively, this 3D model featured a complete crown reconstruction (Fig. 6 B), while partial models obtained by using trajectory control with

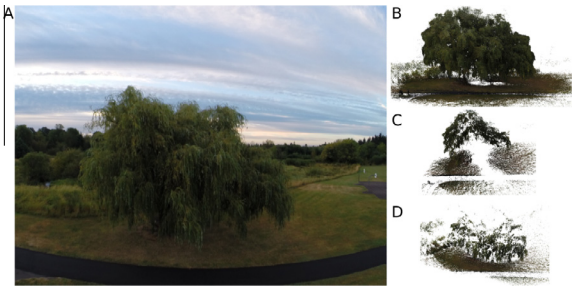

Fig. 6. Approach validation with an actual UAV flight. A: one of the images acquired by the UAV. B: 3D model obtained with active trajectory control enabled. C and D: partial (incomplete) reconstructions obtained without backtracking, shown from the same viewpoint.

lower keypoint matching thresholds featured only a fraction of the crown (Fig. 6 C and D). The single model contained many more points than the sum of the partial models (7,343,810 vs. 1,541,072). Thus, although the partial models could be manually merged into a single model, the resulting model precision would be much lower than the one from the backtracking-enabled reconstruction. Besides, the manual merging of partial models is a laborious and subjective operation.

4. Discussion

Failures in UAV imagery-based 3D scene reconstruction realized during image postprocessing can be costly and irreversible. We have documented that such failures occur because of inadequate number of matched keypoints between images. To guarantee successful 3D reconstruction, we have enabled active UAV trajectory control based on real-time processing of low-resolution imagery acquired during the flight. We have designed an inexpensive UAV configuration amenable to further customizations and used it to test an active control trajectory strategy. The platform can backtrack to and acquire images from locations along its planned trajectory to ensure that an a priori set threshold of matching keypoints is available between adjacent images. We found that this approach offers parsimonious image acquisition while guaranteeing complete scene reconstructions.

Although the techniques presented here have produced complete, gap-free reconstructions of single trees, they may be less successful with more complex vegetation. Owing to occlusion, comprehensive reconstruction of objects with overlapping or interlocking components remains rather challenging, especially for the parts in the object interior. Nevertheless, 3D delineations of dominant trees and other objects present in forested landscapes are invaluable for the assessment and management of natural resources [8] and for crown parameterizations in individual-based forest models [28].

One limitation of the current implementation of active control is the requirement to set beforehand the keypoint matching threshold value. Our experimentation with virtual and real-life environments showed that this threshold is dependent on multiple flight parameters, including camera resolution, illumination conditions and overall scene complexity. In certain conditions, specifying an appropriate matching threshold may not be a major constraint, since it can be derived from experimentation or expert knowledge. In flights with a setup similar to the one used in this study, the same keypoint matching threshold will likely yield successful reconstructions. However, it would be preferable to remove this arbitrary threshold choice. A potential solution could be to perform further processing in real-time to obtain an informed estimate of the probability for a successful 3D reconstruction. In particular, successful completion of bundle adjustment (the simultaneous refinement of the cameras and keypoints locations) would guarantee that one single 3D model will be generated. The feasibility of real-time bundle adjustment using low-cost embedded hardware has not been investigated in this study, and might require computational capacities not available in our current platform.

A general restriction of real-time trajectory control is the computational requirements of visual processing algorithms [12,13,11]. The UAV platform used in this study belongs to a class of so-called mini-scale UAVs with maximum payload of about 1 kg [13], a substantial portion of which is used for batteries. In this context, there are two realistic alternatives: to outsource the computational load to a ground station [19,1], or to perform all processing on embedded hardware, which forces an upper bound on the nature of the image processing used [5,15,11]. In this work, we demonstrated the suitability of a single-board computer with real-time active control that improves 3D reconstruction. Two promising future options for performing the computationally intensive, image-based 3D reconstruction in real-time should be considered: relying on more powerful embedded GPU, or FPGA. With FPGA, the expected execution time gain for SIFT computations is of one order of magnitude [32], with additional potential gains for the sparse and dense reconstruction phases.

References

[1] W.G. Aguilar, C. Angulo, Real-time video stabilization without phantom movements for micro aerial vehicles, EURASIP J. Image Video Process. (1) (2014) 46.
[2] S. Arora, S. Jain, S. Scherer, S. Nuske, L. Chamberlain, S. Singh, Infrastructure-free shipdeck tracking for autonomous landing, in: IEEE International Conference on Robotics and Automation (ICRA), 2013, IEEE, 2013, pp. 323–330.
[3] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (surf), Comput. Vis. Image Understan. 110 (3) (2008) 346–359.
[4] A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, J. Ascenso, R. Cilla, Evaluation of low-complexity visual feature detectors and descriptors, in: 18th International Conference on Digital Signal Processing (DSP), 2013, IEEE, 2013, pp. 1–7.
[5] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti, S. Longhi, A vision-based guidance system for UAV navigation and safe landing using natural landmarks, in: Selected Papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA, 2009, Springer, 2010, pp. 233–257.
[6] E. Chatzilari, G. Liaros, S. Nikolopoulos, Y. Kompatsiaris, A comparative study on mobile visual recognition, in: Machine Learning and Data Mining in Pattern Recognition, Springer, 2013, pp. 442–457.
[7] I. Colomina, P. Molina, Unmanned aerial systems for photogrammetry and remote sensing: a review, ISPRS J. Photogramm. Rem. Sens. 92 (2014) 79–97.
[8] J.W. Coulston, G.G. Moisen, B.T. Wilson, M.V. Finco, W.B. Cohen, C.K. Brewer, et al., Modeling percent tree canopy cover: a pilot study, Photogramm. Eng. Rem. Sens. 78 (7) (2012) 715–727.

[9] J. Cronje, BFROST: binary features from robust orientation segment tests accelerated on the GPU, 2011.
[10] D. Gatziolis, J.F. Lienard, A. Vogs, N.S. Strigul, 3D tree dimensionality assessment using photogrammetry and small unmanned aerial vehicles, PloS one 10 (9) (2015) e0137765.
[11] L. Heng, D. Honegger, G.H. Lee, L. Meier, P. Tanskanen, F. Fraundorfer, M. Pollefeys, Autonomous visual mapping and exploration with a micro aerial vehicle, J. Field Robot. 31 (4) (2014) 654–675.
[12] F. Kendoul, Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems, J. Field Robot. 29 (2) (2012) 315–378.
[13] V. Kumar, N. Michael, Opportunities and challenges with autonomous micro aerial vehicles, Int. J. Robot. Res. 31 (11) (2012) 1279.
[14] D.G. Lowe, Object recognition from local scale-invariant features, The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, vol. 2, IEEE, 1999, pp. 1150–1157.
[15] L. Meier, P. Tanskanen, F. Fraundorfer, M. Pollefeys, Pixhawk: a system for autonomous flight using onboard computer vision, in: IEEE International Conference on Robotics and automation (ICRA), 2011, IEEE, 2011, pp. 2992–2997.
[16] L. Meier, P. Tanskanen, L. Heng, G.H. Lee, F. Fraundorfer, M. Pollefeys, Pixhawk: a micro aerial vehicle design for autonomous flight using onboard computer vision, Auton. Robot. 33 (1–2) (2012) 21–39.
[17] D. Mellinger, N. Michael, V. Kumar, Trajectory generation and control for precise aggressive maneuvers with quadrotors, Int. J. Robot. Res. (2012). 0278364911434236.
[18] I.F. Mondragón, P. Campoy, C. Martinez, M.A. Olivares-Méndez, 3d pose estimation based on planar object tracking for UAVs control, in: IEEE International Conference on Robotics and Automation (ICRA), 2010, 2010, pp. 35.
[19] T. Mori, S. Scherer, First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles, in: IEEE International Conference on Robotics and Automation (ICRA), 2013, IEEE, 2013, pp. 1750–1757.
[20] P. Paleo, E. Pouyet, J. Kieffer, Image stack alignment in full-field X-ray absorption spectroscopy using SIFT_PyOCL, J. Synchrotron Radiat. 21 (2) (2014).
[21] R. Roberts, D.-N. Ta, J. Straub, K. Ok, F. Dellaert, Saliency detection and model-based tracking: a two part vision system for small robot navigation in forested environment, in: SPIE Defense, Security, and Sensing, page 83870S. International Society for Optics and Photonics, 2012.
[22] S. Ross, N. Melik-Barkhudarov, K.S. Shankar, A. Wendel, D. Dey, J.A. Bagnell, M. Hebert, Learning monocular reactive UAV control in cluttered natural environments, in: IEEE International Conference on Robotics and Automation (ICRA), 2013, IEEE, 2013, pp. 1765–1772.
[23] M. Rothermel, K. Wenzel, D. Fritsch, N. Haala, Sure: photogrammetric surface reconstruction from imagery, in: Proceedings LC3D Workshop, Berlin, 2012.
[24] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, Orb: an efficient alternative to sift or surf, in: IEEE International Conference on Computer Vision (ICCV), 2011, IEEE, 2011, pp. 2564–2571.
[25] F. Ruffier, N. Franceschini, Optic flow regulation in unsteady environments: a tethered MAV achieves terrain following and targeted landing over a moving platform, J. Intell. Robot. Syst. (2014) 1–19.
[26] M. Sauerbier, H. Eisenbeiss, UAVs for the documentation of archaeological excavations, Int. Arch. Photogramm. Rem. Sens. Spatial Inform. Sci. (Part 5) (2010).
[27] S. Shen, Y. Mulgaonkar, N. Michael, V. Kumar, Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor, in: Robotics: Science and Systems. Citeseer, 2013.
[28] N. Strigul, Individual-based models and scaling methods for ecological forestry: implications of tree phenotypic plasticity, in: J. Garcia, J. Casero (Eds.), Sustainable Forest Management, InTech, Rijeka, Croatia, 2012, pp. 359–384.
[29] C. Wu, Towards linear-time incremental structure from motion, in: International Conference on 3D Vision-3DV 2013, 2013, IEEE, 2013, pp. 127–134.
[30] B. Yang, C. Chen, Automatic registration of UAV-borne sequent images and LiDAR data, ISPRS J. Photogramm. Rem. Sens. 101 (2015) 262–274.
[31] S. Yang, S.A. Scherer, A. Zell, Visual SLAM for autonomous MAVs with dual cameras, in: IEEE International Conference on Robotics and Automation (ICRA), 2014, IEEE, 2014, pp. 5227–5232.
[32] L. Yao, H. Feng, Y. Zhu, Z. Jiang, D. Zhao, W. Feng, An architecture of optimised SIFT feature detection for an FPGA implementation of an image matcher, in: International Conference on Field-Programmable Technology, 2009, FPT 2009, IEEE, 2009, pp. 30–37.