Markerless Visual Control of a Quad-Rotor Micro Aerial Vehicle by Means of On-Board Stereo Processing

Konstantin Schauwecker[1], Nan Rosemary Ke, Sebastian Andreas Scherer[1], and Andreas Zell[1]

[1] University of Tübingen, Wilhelm-Schickard-Institute for Computer Science, Department Cognitive Systems, Sand 1, 72076 Tübingen, Germany

Abstract. We present a quad-rotor micro aerial vehicle (MAV) that is capable to fly and navigate autonomously in an unknown environment. The only sensory input used by the MAV are the imagery from two cameras in a stereo configuration, and data from an inertial measurement unit. We apply a fast sparse stereo matching algorithm in combination with a visual odometry method based on PTAM to estimate the current MAV pose, which we require for autonomous control. All processing is performed on a single board computer on-board the MAV. To our knowledge, this is the first MAV that uses stereo vision for navigation, and does not rely on visual markers or off-board processing. In a flight experiment, the MAV was capable to hover autonomously, and it was able to estimate its current position at a rate of 29 Hz and with an average error of only 2.8cm.

1 Introduction

Micro aerial vehicles (MAV) have recently received much attention in the research community. Improvements in sensors, propulsion and processing capabilities have promoted the development of MAVs that are able to fly and navigate autonomously. The key advantage of MAVs when compared to other aircrafts is their relatively small size, which allows them to be operated inside buildings or other confined spaces. This is particularly true for rotary-wing MAVs that are able to hover and have a high maneuverability.

Indoor environments are, however, very challenging for autonomous navigation due to the lack of GPS coverage. Hence, an autonomous MAV needs to be equipped with other sensors. The choice of appropriate sensors is limited by several factors, such as the low maximum payload, power consumption and processing resources. In this paper we present a quad-rotor MAV capable of autonomous indoor flight, which uses two synchronized cameras in a stereo configuration. The only other sensor is an inertial measurement unit (IMU), as commonly found on many MAVs.

Although the cameras we use are light and have low power consumption, performing stereo analysis and motion estimation by means of visual odometry is very computationally intensive. Thus, the MAV has to provide sufficient processing power. Our quad-rotor is based on the open source and open hardware MAV platform PIXHAWK that has been developed by the ETH Zürich [MTH+12]. In its reference design, this quad-rotor features a powerful single board computer with a 1.8 GHz dual-core CPU.

Although the PIXHAWK on-board computer provides much more processing power than is commonly found on MAVs of comparable size, using it for performing stereo

matching and visual odometry in real time is still challenging. In fact, to our knowledge there have been no other successful attempts of stereo vision guided MAVs that do not depend on visual markers or a ground station that performs most of the required processing. Hence, the presented quad-rotor is the first truly autonomous MAV that relies on stereo vision and can operate in an arbitrary and unknown environment.

2 Related Work

For the autonomous control of an MAV, we are required to estimate its location and movement. We can obtain such an estimate from the data provided by an IMU, however, IMU readings are prone to drift. Alternatively, a camera can be used for visual motion estimation, known as visual odometry. The difficulty with this method is, however, that a single camera can only provide estimates relative to an unobservable scaling factor. Hence, most monocular vision based approaches only work in specific environments.

For example, in [TVHF06] visual markers are used for an autonomous quad-rotor MAV that relies on monocular vision. Processing, however, is not performed onboard, but the camera image is transmitted wirelessly to a computer that remotely controls the MAV. An approach with on-board processing can be found in [WZ09], where an infrared camera extracted from a Wii Mote is used to track a known configuration of infrared LEDs. A method that does not require any visual markers and only relies on on-board processing was published in [AAWS11]. The authors use a visual SLAM algorithm with a low update rate of about 10 Hz. The unknown scaling factor is estimated using readings from an accelerometer and pressure sensor.

The dependency on visual markers or scaling factor estimation can be avoided when using stereo vision. Due to the computational demands of stereo matching, most existing work has focused on off-board processing. In [PKA+10] and [AZKB09] a ground-mounted stereo camera has been used that is focused on a quad-rotor MAV. A computer performs the stereo processing and tracking, and sends remote control signals. In contrast, an on-board mounted stereo camera has been used in [CLLP12], which transmits images to a ground computer that performs the image processing and quad-rotor control. This transmission happens at a relatively low rate of just 13 Hz. The system then extracts about 150 features and uses a KLT tracker for performing visual odometry.

The are few publications of MAVs capable of on-board stereo processing. One example is the quad-rotor presented in [HMT+11], which is also based on the PIXHAW platform. This MAV runs a block-matching based stereo algorithm at an image resolution of 320×240 that was extended to 640×480 in [MTH+12]. The stereo matching results are used for an obstacle avoidance algorithm running at 15 Hz. Despite the on-board stereo matching, this MAV still requires visual markers for tracking its position.

To achieve markerless autonomous control by means of stereo vision and on-board processing, we require a fast method for stereo matching and visual odometry. Recently, a very fast and accurate sparse stereo matching algorithm has been published in [SKZ12]. This method achieves high processing rates by employing the optimized implementation of the FAST feature detector presented in [SZ12]. Due to its high speed and good accuracy, this method seems ideal for stereo processing on-board an MAV.

One efficient system for visual motion and location estimation that has received a lot of attention is Parallel Tracking and Mapping (PTAM) [KM07]. This method not only

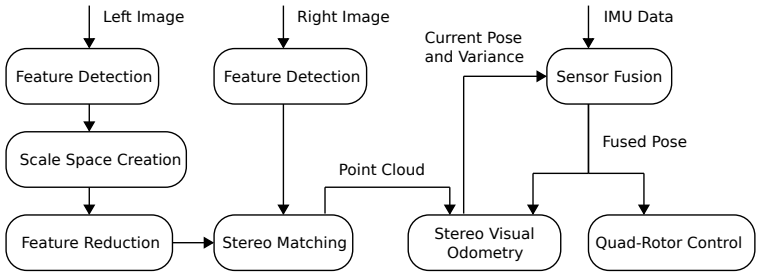


Fig. 1: System design of the on-board navigation software.

performs visual odometry, but full simultaneous localization and mapping (SLAM). The authors have made their source code available, which makes this a great starting point for creating new visual odometry or SLAM algorithms. One such adaptation of PTAM was recently published in [SDZ12], which uses depth information from an RGB-D camera. The same method can be applied to data gathered through stereo vision.

3 Method

The system design of the presented quad-rotor MAV is visualized in Fig. 1. Our processing pipeline computes an estimate for the quad-rotor pose, based on the stereo imagery and IMU data. This pose estimate is used as input by the control software that runs on a separate micro controller. For the control software, we use the unmodified PID controller provided by the PIXHAWK project, and hence we will not discuss this part.

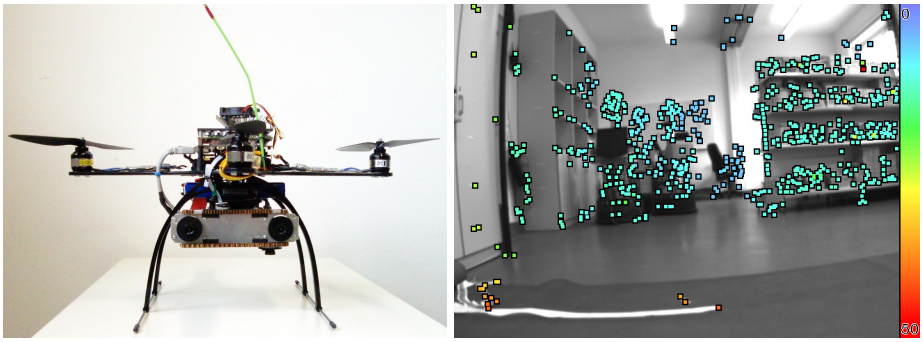
3.1 Feature Detection and Stereo Matching

Our quad-rotor has been equipped with two grayscale USB cameras, which are operated with a resolution of 640×480 at 30 Hz. As can be seen in Fig. 2a, the cameras are mounted in a forward facing stereo configuration with a baseline of 11 cm. The cameras are synchronized using their built-in external trigger and strobe functionality.

Our stereo matching system is based on the previously introduced feature detector and stereo matching algorithm from [SKZ12]. The fact that this method only provides a sparse set of matches is not a limitation in our case, as we will be using the results for feature-based visual odometry. For this task, however, we require a scale space for supporting feature re-detecting. As processing time is extremely critical for our system, we create this scale space without re-running the feature detection on each scale level.

Instead, in each scale level we only evaluate those pixels, for which a feature was detected at the preceding level. Thus, a feature detected in any scale level can be traced back to a feature from the primary level, and we only have to retain the maximum level l up to which the feature has been detected. This method also has the advantage that it provides sub-pixel accuracy for all scale levels except the primary one.

The number of features that are detected is crucial for the system performance. In case of too many features, it will be impossible to meet the desired processing rate. If, on the other hand, too few features are detected, then the system will only achieve a



(a) (b)

Fig. 2: (b) Our quad-rotor MAV equipped with two cameras in a stereo configuration and (b) performance example of our stereo matching system.

poor accuracy. We solve this predicament by employing a reasonable detection threshold and applying an additional feature reduction step to the left image, if too many features are detected. Features detected in the right image are left unchanged, as the feature reduction could eliminate different features for both images and thus reduce the matching performance. We aim at reducing the number of detected features to 1000.

The advantage of the employed feature detector is its less clustered feature distribution, which we would like to keep. We hence reduce the detected features in a way that retains the spatial distribution. For this purpose, we first calculate the percentage p of the detected features, which we want to remove. We then divide the input image into a grid of 5×4 rectangular cells, and for each cell we create a list of the contained features. This list is sorted in descending order by the maximum detected scale level l. In case of identical l, we use the feature score s for determining their sort order.

The list of features is then shortened from the end by the percentage p. We keep an error term to ensure that rounding errors do not lead to too many or too few eliminated features. Giving preference to features that have been detected on many scale levels is of advantage for the visual odometry algorithm. An example for the performance of the resulting stereo matching system during autonomous flight is given in Fig. 2b.

3.2 Visual Odometry

For performing the visual motion estimation, we employ the previously discussed PTAM extension published in [SDZ12]. While this adaptation originally used an RGB-D camera to obtain depth values for the detected image features, we instead use the depth information gained by our stereo matching system for successfully matched features. Unlike in [SDZ12], we do not further process features for which no depth information is available, as we are not facing the same coverage problems as the RGB-D camera.

The problem with this motion estimation method is that it only works well for cases where the camera motion is constrained to a small volume. Otherwise, the system will keep creating new keyframes and mapping new features, which quickly degrades the

overall performance. Due to the missing loop closure detection, this can also happen without large camera displacements, as the system will keep expanding the map if it is unable to re-detect previously mapped features due to accumulated errors.

This is a severe limitation, as a quad-rotor is capable to quickly cover a long distance and requires frequent pose estimates to achieve stable control. To overcome this problem, we simplify PTAM by avoiding the global optimization step and discarding all keyframes that are no longer considered for local optimization. This way, the map no longer grows and we should be able to achieve a constant performance. However, this means that the resulting method is no longer performing full SLAM, as only a small map required for local optimization is retained. In fact, this approach is now performing visual odometry. For our purposes, however, a visual odometry algorithm is sufficient and we opt in favor of the faster and more predictable processing speed.

We performed one further modification of PTAM, which is the replacement of the lens distortion model. The original model was based on an arctangent transformation, which we replaced with the more accurate and commonly used Brown's distortion model [Bro66]. Because we perform stereo matching, we require a more accurate camera calibration, which is why the original camera model was insufficient.

3.3 Sensor Fusion

We fuse the data from visual odometry and IMU with an extended Kalman filter. For this purpose, we use the open source ROS package *imu_filter*[1], which already provides an appropriate Kalman Filter implementation. This filter uses the IMU sensor data for performing the Kalman prediction step. For every pose estimate that is obtained through visual odometry, the filter then calculates the Kalman correction.

As input for the Kalman filter, we also need the pose variance. For the position, the variance can easily be obtained from the least-squares optimization performed by PTAM, but for the rotation this is more difficult. This is because PTAM represents rotations with a vector $\boldsymbol{v} \in \mathbb{R}^3$ that indicates the rotation axis and whose length $\alpha = ||\boldsymbol{v}||$ determines the rotation angle. Both, *imu_filter* and the control software represent rotations with quaternions, which consist of a normalized rotation axis $\boldsymbol{u}$ and a value w, indicating the rotation amount. In our case they can be determined as follows:

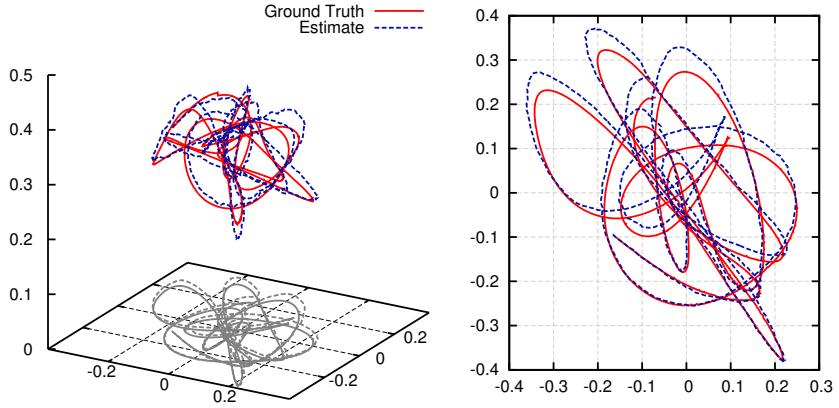
$$\boldsymbol{u} = \boldsymbol{v} \cdot \frac{1}{||\boldsymbol{v}||} \cdot \sin\frac{\alpha}{2} \tag{1}$$

$$w = cos\frac{\alpha}{2} \tag{2}$$

If we assume a normal distribution, we can approximate the new standard deviation using error analysis. We start with estimating the standard deviation σ_α of the rotation angle α. By applying the error propagation rules for sums and exponents to the squares, sum and square root that are required for calculating the vector norm, we receive:

$$\sigma_\alpha \approx \frac{||\boldsymbol{v}\boldsymbol{\sigma}_{\boldsymbol{v}}||}{\alpha} \tag{3}$$

[1] Available form: `http://ros.org/wiki/imu_filter`



(a) (b)

Fig. 3: (a) Perspective and (b) top-view of the ground truth motion information and on-board motion estimates. The scale of both diagrams is in meters.

For finding the standard deviations for Eq.(1) and (2), we need to estimate the error propagation of the sine and cosine function, for which we use the approximations

$$f = \sin x \,; \sigma_f^2 \approx \sigma_x^2 \cos^2 x \quad \text{and} \quad f = \cos x \,; \sigma_f^2 \approx \sigma_x^2 \sin^2 x \,.$$

With these approximations, Eq (3) and the error propagation rule for products, we receive the following standard deviations, with which we perform the sensor fusion:

$$\boldsymbol{\sigma_u} \approx \boldsymbol{\sigma_v} \cdot \frac{1}{2} \cdot \left| \cos \frac{\alpha}{2} \right|$$
$$\sigma_w \approx \frac{\sigma_\alpha}{2} \cdot \left| \sin \frac{\alpha}{2} \right|$$

The fused pose is passed to the control software and back to our PTAM-based visual odometry method, to improve motion prediction accuracy. We have extended the PTAM motion model to allow feedback from the sensor fusion, by always using the fused pose as the pose estimate for the previous frame. The IMU data provides an absolute measurements for the roll and pitch angles. By feeding back the fused pose, we avoid roll and pitch drifts, which could have severe implications on the quad-rotor stability.

4 Experiments and Evaluation

In repeated successful experiments, we have verified that the presented quad-rotor is capable to fly autonomously. In this section, we will examine one such test run recorded in an indoor setting. The quad-rotor was supposed to hover autonomously at a fixed position, while take-off and landing was performed manually. The total flight time was 45.7s, of which we neglect the first 8.6s and the last 3.1s for take-off and landing. This leaves us an autonomous flight time of 33.9s to analyze.

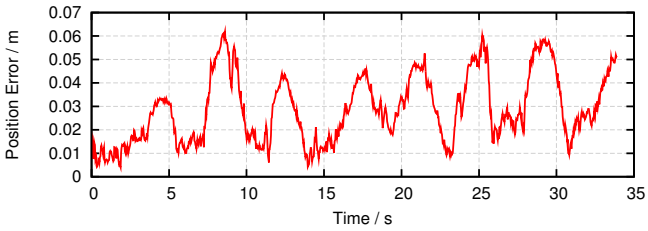


Fig. 4: Error of the on-board position estimation.

Process	CPU Usage
Stereo Matching	37.4%
Visual Odometry	32.5%
Image Acquisition	6.7%
Data Recording	5.8%
Sensor Fusion	1.1%
Other	2.5%
Total	**85.6%**

Tab. 1: CPU usage.

We recorded all sensor data and the outcome of the on-board pose estimation. Further, we also recorded ground truth motion information with an Optitrack tracking system. We aligned the on-board pose estimates to the ground truth, such that they closely match when autonomous hovering starts. The aligned motion data is shown in Fig. 3.

In total, the cameras recorded 1019 frames each, and our processing pipeline was able to generate 29.3 pose estimates per second on average. This is very close to the camera frame rate of 30 Hz, indicating that only very few frames have been dropped. The average number of detected features in the left image was 999.5, of which 64% were successfully matched during stereo processing. Further, we recorded CPU load statistics, which are given in Tab. 1. The table shows that if we had omitted the data recording, the CPU usage would have been below 80%.

To measure the performance, we examine how well our MAV can keep its hovering location. As target position, we consider the average position of the evaluated time span. Using the recorded ground truth, we receive an average error of 0.26m, and a root mean square error (RMSE) of 0.32m, which should be low enough for indoor applications.

For measuring the accuracy of the on-board pose estimates, we calculate the Euclidean distance between the pose estimates and the ground truth, which we have done in Fig. 4. This figure reveals that the error stayed bounded throughout the hovering experiment. In fact, the average error is only 2.8cm, and the RMSE has a value of 3.2cm, which is an order of magnitude less than the error of the hovering position.

5 Conclusions

In this paper we presented an autonomous quad-rotor MAV based on the PIXHAWK platform. Our MAV is capable to fly autonomously by only using a stereo camera and an IMU as sensors. To our knowledge, we are the first to achieve this goal without relying on visual markers or off-board image processing.

The key challenge in creating this MAV were the limited on-board computing resources. By combining and adapting efficient methods for stereo matching and visual odometry, we have been able to create a high-performance processing pipeline for estimating the current MAV pose. In fact, this processing pipeline has been able to provide pose updates at a rate of 29 Hz, without even using all CPU resources. Hence, we could even use more demanding parameterizations or perform further high-level tasks.

We conducted an autonomous hovering experiment for which we gathered accurate ground truth with a tracking system. According to our analysis, the MAV was able to

keep the desired hovering position with an average error of 0.26m. The on-board pose estimation was able to operate with an average error of just 2.8cm. Both values are low enough to facilitate safe and autonomous navigation of the MAV inside buildings.

In future work, we will be focusing on using the presented MAV to perform more complex autonomous navigation tasks. Further, our aim is to build an MAV that interacts with its environment. This would be possible by exhausting the 3D range data we receive from the employed stereo matching method, with which we could facilitate applications such as autonomous exploration, path planning or obstacle avoidance.

References

[AAWS11] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart. Onboard IMU and monocular vision based control for MAVs in unknown in-and outdoor environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3056–3063, 2011.

[AZKB09] M. Achtelik, T. Zhang, K. Kuhnlenz, and M. Buss. Visual Tracking and Control of a Quadcopter Using a Stereo Camera System and Inertial Sensors. In *International Conference on Mechatronics and Automation (ICMA)*, pages 2863–2869. IEEE, 2009.

[Bro66] D. C. Brown. Decentering Distortion of Lenses. *Photogrammetric Engineering*, 7:444–462, 1966.

[CLLP12] L. R. G. Carrillo, A. E. D. López, R. Lozano, and C. Pégard. Combining Stereo Vision and Inertial Navigation System for a Quad-Rotor UAV. *Journal of Intelligent & Robotic Systems*, 65(1):373–387, 2012.

[HMT+11] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Autonomous Obstacle Avoidance and Maneuvering on a Vision-Guided MAV Using On-Board Processing. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2472–2477, 2011.

[KM07] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 1–10, 2007.

[MTH+12] L. Meier, P. Tanskanen, L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys. PIXHAWK: A Micro Aerial Vehicle Design for Autonomous Flight Using Onboard Computer Vision. *Autonomous Robots*, pages 1–19, 2012.

[PKA+10] D. Pebrianti, F. Kendoul, S. Azrad, W. Wang, and K. Nonami. Autonomous Hovering and Landing of a Quad-Rotor Micro Aerial Vehicle by Means of on Ground Stereo Vision System. *Journal of System Design and Dynamics*, 4(2):269–284, 2010.

[SDZ12] S. A. Scherer, D. Dube, and A. Zell. Using Depth in Visual Simultaneous Localisation and Mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. Forthcomming.

[SKZ12] K. Schauwecker, R. Klette, and A. Zell. A New Feature Detector and Stereo Matching Method for Accurate High-Performance Sparse Stereo Matching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012. Submitted.

[SZ12] K. Schauwecker and A. Zell. Faster FAST Corners by Using SIMD Instructions. In *European Conference on Computer Vision (ECCV)*, 2012. Submitted.

[TVHF06] G. P. Tournier, M. Valenti, J.P. How, and E. Feron. Estimation and Control of a Quadrotor Vehicle Using Monocular Vision and Moiré Patterns. In *In AIAA Guidance, Navigation and Control Conference*, pages 2006–6711, 2006.

[WZ09] K. E. Wenzel and A. Zell. Low-Cost Visual Tracking of a Landing Place and Hovering Flight Control with a Microcontroller. In *International Symposium on Unmanned Aerial Vehicles (UAV)*, pages 1–15. Kimon P. Valavanis, 2009.