Accurate Figure Flying with a Quadrocopter Using Onboard Visual and Inertial Sensing

Jakob Engel, Jürgen Sturm, Daniel Cremers

Abstract— **We present an approach that enables a low-cost quadrocopter to accurately fly various figures using vision as main sensor modality. Our approach consists of three components: a monocular SLAM system, an extended Kalman filter for data fusion and state estimation and a PID controller to generate steering commands. Our system is able to navigate in previously unknown indoor and outdoor environments at absolute scale without requiring artificial markers or external sensors. Next to a full description of our system, we introduce our scripting language and present several examples of accurate figure flying in the corresponding video submission.**

I. Introduction

In recent years, research interest in autonomous micro-aerial vehicles (MAVs) has grown rapidly. Significant progress has been made, recent examples include aggressive flight maneuvers [1, 2], ping-pong [3] and collaborative construction tasks [4]. However, all of these systems require external motion capture systems. Flying in unknown, GPS-denied environments is still an open research problem. The key challenges here are to localize the robot purely from its own sensor data and to robustly navigate it even under potential sensor loss. This requires both a solution to the so-called simultaneous localization and mapping (SLAM) problem as well as robust state estimation and control methods. These challenges are even more expressed on low-cost hardware with inaccurate actuators, noisy sensors, significant delays and limited onboard computation resources.

For solving the SLAM problem on MAVs, different types of sensors such laser range scanners [5], monocular cameras [6, 7], stereo cameras [8] and RGB-D sensors [9] have been explored in the past. In our point of view, monocular cameras provide two major advantages above other modalities: (1) the amount of information that can be acquired is immense compared to their low weight, power consumption, size and cost, which are unmatched by any other type of sensor and (2) in contrast to depth measuring devices, the range of a monocular camera is virtually unlimited – allowing a monocular SLAM system to operate both in small, confined and large, open environments. The drawback however is, that the scale of the environment cannot be determined from monocular vision alone, such that additional sensors (such as an IMU) are required.

The motivation behind our work is to showcase that robust, scale-aware visual navigation is feasible and safe on low-cost robotic hardware. As a platform, we use the Parrot AR.Drone which is available for $300 and, with a weight of only 420 g and a protective hull, safe to be used in public places. As the onboard computational resources are utterly limited, all computations are performed externally.

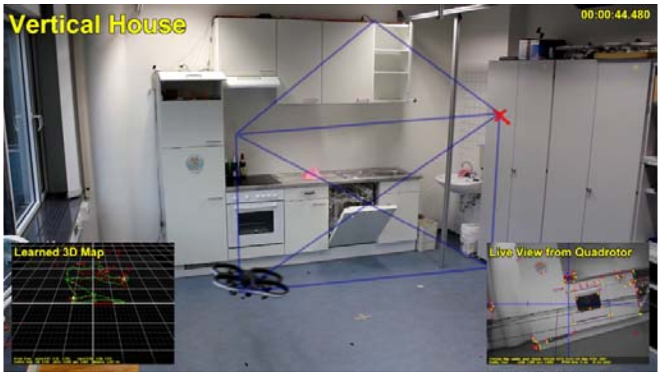


Fig. 1. Our approach enables a low-cost quadrocopter to accurately follow any given flight trajectory. We use the front-facing camera as the main sensor for localization based on PTAM [11]. **Middle:** Flight figure (blue lines) and current goal location (red cross). **Bottom left:** Learned 3D feature map. **Bottom right:** Visual features detected and the live-stream from the quadrocopter.

This paper is an extension of our recently published work [10]. In this work, we additionally describe the scripting language that enables our quadrocopter to complete complex flight patterns including take-off, autonomous map initialization, and landing. This paper comes with two videos, demonstrating the robustness of our approach, its ability to eliminate drift effectively and to follow pre-defined, absolute scale trajectories. They are available online at:

`http://youtu.be/tZxlDly7lno`
`http://youtu.be/eznMokFQmpc`

II. Related Work

Previous work on autonomous flight with quadrocopters can be categorized into different research areas. One part of the community focuses on accurate quadrocopter control and a number of impressive results have been published [12, 1, 3]. These works however rely on advanced external tracking systems, restricting their use to a lab environment. A similar approach is to distribute artificial markers in the environment, simplifying pose estimation [13]. Other approaches learn a map offline from a previously recorded, manual flight and thereby enable a quadrocopter to again fly the same trajectory [14]. For outdoor flights where GPS-based pose estimation is possible, complete solutions are available as commercial products [15].

In this work we focus on autonomous flight without previous knowledge about the environment nor GPS signals, while

J. Engel, J. Sturm and D. Cremers are with the Department of Computer Science, Technical University of Munich, Germany {engelj,sturmju,cremers}@in.tum.de

using only onboard sensors. First results towards this goal have been presented using a lightweight laser scanner [5], a Kinect [9] or a stereo rig [8] mounted on a quadrocopter as primary sensor. While these sensors provide absolute scale of the environment, their drawback is a limited range and large weight, size and power consumption when compared to a monocular setup [16, 7].

In our work we therefore focus on a monocular camera for pose estimation. Stabilizing controllers based on optical flow were presented in [17], and similar methods are integrated in commercially available hardware [18]. Such systems however are subject to drift over time, and hence not suited for long-term navigation.

To eliminate drift, various monocular SLAM methods have been investigated on quadrocopters, both with off-board [16, 5] and on-board processing [7]. A particular challenge for monocular SLAM is, that the scale of the map needs to be estimated from additional metric sensors such as IMU or GPS, as it cannot be recovered from vision alone. This problem has been addressed in recent publications such as [19, 20]. The current state of the art is to estimate the scale using an extended Kalman filter (EKF), which contains scale and offset in its state. In contrast to this, we propose a novel approach which is based on direct computation: Using a statistical formulation, we derive a closed-form, consistent estimator for the scale of the visual map. Our method yields accurate results both in simulation and practice, and requires less computational resources than filtering. It can be used with any monocular SLAM algorithm and sensors providing metric position or velocity measurements, such as an ultrasonic or pressure altimeter or occasional GPS measurements.

In contrast to the systems presented in [16, 7], we deliberately refrain from using expensive, customized hardware: the only hardware required is the AR.Drone, which comes at a costs of merely $300 – a fraction of the cost of quadrocopters used in previous work. Released in 2010 and marketed as high-tech toy, it has been used and discussed in several research projects [21, 22, 23]. To our knowledge, we are the first to present a complete implementation of autonomous, camera-based flight in unknown, unstructured environments using the AR.Drone.

III. Hardware Platform

As platform we use the Parrot AR.Drone, a commercially available quadrocopter. Compared to other modern MAV's such as Ascending Technology's Pelican or Hummingbird quadrocopters, its main advantage is the very low price, its robustness to crashes and the fact that it can safely be used indoor and close to people. This however comes at the price of flexibility: Neither the hardware itself nor the software running onboard can easily be modified, and communication with the quadrocopter is only possible over wireless LAN. With battery and hull, the AR.Drone measures 53 cm × 52 cm and weights 420 g.

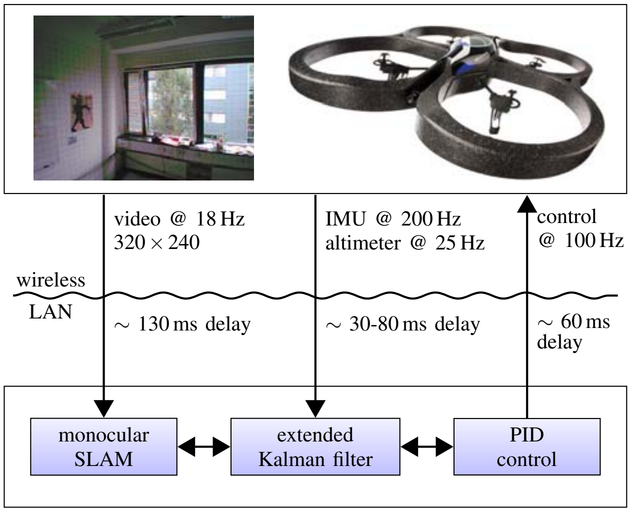


Fig. 2. Approach Outline: Our navigation system consists of three major components: a monocular SLAM implementation for visual tracking, an EKF for data fusion and prediction, and PID control for pose stabilization and navigation. All computations are performed offboard, which leads to significant, varying delays which our approach has to compensate.

A. *Sensors*

The AR.Drone is equipped with a 3-axis gyroscope and accelerometer, an ultrasound altimeter and two cameras. The first camera is aimed forward, covers a field of view of $73.5^\circ \times 58.5^\circ$, has a resolution of 320×240 and a rolling shutter with a delay of 40 ms between the first and the last line captured. The video of the first camera is streamed to a laptop at 18 fps, using lossy compression. The second camera aims downward, covers a field of view of $47.5^\circ \times 36.5^\circ$ and has a resolution of 176×144 at 60 fps. The onboard software uses the down-looking camera to estimate the horizontal velocity. The quadcopter sends gyroscope measurements and the estimated horizontal velocity at 200 Hz, the ultrasound measurements at 25 Hz to the laptop. The raw accelerometer data cannot be accessed directly.

B. *Control*

The onboard software uses these sensors to control the roll Φ and pitch Θ, the yaw rotational speed $\dot{\Psi}$ and the vertical velocity $\dot{z}$ of the quadrocopter according to an external reference value. This reference is set by sending a new control command $\mathbf{u} = (\bar{\Phi}, \bar{\Theta}, \bar{\dot{z}}, \bar{\dot{\Psi}}) \in [-1,1]^4$ every 10 ms.

IV. Approach

Our approach consists of three major components running on a laptop connected to the quadrocopter via wireless LAN, an overview is given in Fig. 2.

1) **Monocular SLAM:** For monocular SLAM, our solution is based on Parallel Tracking and Mapping (PTAM) [11]. After map initialization, we rotate the visual map such that the *xy*-plane corresponds to the horizontal plane according to the accelerometer data, and scale it such that the average keypoint depth is 1. Throughout tracking, the scale of the map $\lambda \in \mathbb{R}$ is estimated using a novel method described in Section IV-A. Furthermore, we use the pose estimates from the EKF to identify and reject falsely tracked frames.

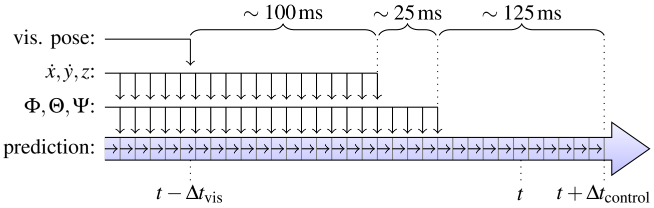


Fig. 3. Pose Prediction: Measurements and control commands arrive with significant delays. To compensate for these delays, we keep a history of observations and sent control commands between $t - \Delta t_{\mathrm{vis}}$ and $t + \Delta t_{\mathrm{control}}$ and re-calculate the EKF state when required. Note the large timespan with no or only partial odometry observations.

2) **Extended Kalman Filter:** In order to fuse all available data, we employ an extended Kalman filter (EKF). We derived and calibrated a full motion model of the quadrocopter's flight dynamics and reaction to control commands, which we will describe in more detail in Section IV-B. This EKF is also used to compensate for the different time delays in the system, arising from wireless LAN communication and computationally complex visual tracking.

We found that height and horizontal velocity measurements arrive with the same delay, which is slightly larger than the delay of attitude measurements. The delay of visual pose estimates Δt_{vis} is by far the largest. Furthermore we account for the time required by a new control command to reach the drone $\Delta t_{\mathrm{control}}$. All timing values given subsequently are typical values for a good connection, the exact values depend on the wireless connection quality and are determined by a combination of regular ICMP echo requests sent to the quadrocopter and calibration experiments.

Our approach works as follows: first, we time-stamp all incoming data and store it in an observation buffer. Control commands are then calculated using a prediction for the quadrocopter's pose at $t + \Delta t_{\mathrm{control}}$. For this prediction, we start with the saved state of the EKF at $t - \Delta t_{\mathrm{vis}}$ (i.e., after the last visual observation/unsuccessfully tracked frame). Subsequently, we predict ahead up to $t + \Delta t_{\mathrm{control}}$, using previously issued control commands and integrating stored sensor measurements as observations. This is illustrated in Fig. 3. With this approach, we are able to compensate for delayed and missing observations at the expense of recalculating the last cycles of the EKF.

3) **PID Control:** Based on the position and velocity estimates from the EKF at $t + \Delta t_{\mathrm{control}}$, we apply PID control to steer the quadrocopter towards the desired goal location $\mathbf{p} = (\hat{x}, \hat{y}, \hat{z}, \hat{\Psi})^T \in \mathbb{R}^4$ in a global coordinate system. According to the state estimate, we rotate the generated control commands to the robot-centric coordinate system and send them to the quadrocopter. For each of the four degrees-of-freedom, we employ a separate PID controller for which we experimentally determined suitable controller gains.

A. Scale Estimation

One of the key contributions of this paper is a closed-form solution for estimating the scale $\lambda \in \mathbb{R}^+$ of a monocular SLAM system. For this, we assume that the robot is able to make noisy measurements of absolute distances or velocities from additional, metric sensors such as an ultrasound altimeter.

As a first step, the quadrocopter measures in regular intervals the d-dimensional distance traveled both using only the visual SLAM system (subtracting start and end position) and using only the metric sensors available (subtracting start and end position, or integrating over estimated speeds). Each interval gives a pair of samples $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{R}^d$, where $\mathbf{x}_i$ is scaled according to the visual map and $\mathbf{y}_i$ is in metric units. As both $\mathbf{x}_i$ and $\mathbf{y}_i$ measure the motion of the quadrocopter, they are related according to $\mathbf{x}_i \approx \lambda \mathbf{y}_i$.

More specifically, if we assume Gaussian noise in the sensor measurements with constant variance[1], we obtain

$$\mathbf{x}_i \sim \mathcal{N}(\lambda \boldsymbol{\mu}_i, \sigma_x^2 \mathbf{I}_{3\times 3}) \tag{1}$$

$$\mathbf{y}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \sigma_y^2 \mathbf{I}_{3\times 3}) \tag{2}$$

where the $\boldsymbol{\mu}_i \in \mathbb{R}^d$ denote the true (unknown) distances covered and $\sigma_x^2, \sigma_y^2 \in \mathbb{R}^+$ the variances of the measurement errors. Note that the individual $\boldsymbol{\mu}_i$ are not constant but depend on the actual distances traveled by the quadrocopter in the measurement intervals.

One possibility to estimate λ is to minimize the sum of squared differences (SSD) between the re-scaled measurements, i.e., to compute one of the following:

$$\lambda_y^* := \arg\min_{\lambda} \sum_i \|\mathbf{x}_i - \lambda \mathbf{y}_i\|^2 = \frac{\sum_i \mathbf{x}_i^T \mathbf{y}_i}{\sum_i \mathbf{y}_i^T \mathbf{y}_i} \tag{3}$$

$$\lambda_x^* := \left(\arg\min_{\lambda} \sum_i \|\lambda \mathbf{x}_i - \mathbf{y}_i\|^2 \right)^{-1} = \frac{\sum_i \mathbf{x}_i^T \mathbf{x}_i}{\sum_i \mathbf{x}_i^T \mathbf{y}_i}. \tag{4}$$

The difference between these two lines is whether one aims at scaling the $\mathbf{x}_i$ to the $\mathbf{y}_i$ or vice versa. However, both approaches lead to different results, none of which converges to the true scale λ when adding more samples. To resolve this, we propose a maximum likelihood (ML) approach, that is estimating λ by minimizing the negative log-likelihood

$$\mathcal{L}(\boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_n, \lambda) \propto \frac{1}{2} \sum_{i=1}^{n} \left(\frac{\|\mathbf{x}_i - \lambda \boldsymbol{\mu}_i\|^2}{\sigma_x^2} + \frac{\|\mathbf{y}_i - \boldsymbol{\mu}_i\|^2}{\sigma_y^2} \right) \tag{5}$$

By first minimizing over the $\boldsymbol{\mu}_i$ and then over λ, it can be shown analytically that (5) has a unique, global minimum at

$$\boldsymbol{\mu}_i^* = \frac{\lambda^* \sigma_y^2 \mathbf{x}_i + \sigma_x^2 \mathbf{y}_i}{\lambda^{*2} \sigma_y^2 + \sigma_x^2} \tag{6}$$

$$\lambda^* = \frac{s_{xx} - s_{yy} + \operatorname{sign}(s_{xy}) \sqrt{(s_{xx} - s_{yy})^2 + 4 s_{xy}^2}}{2 \sigma_x^{-1} \sigma_y s_{xy}} \tag{7}$$

with $s_{xx} := \sigma_y^2 \sum_{i=1}^n \mathbf{x}_i^T \mathbf{x}_i$, $s_{yy} := \sigma_x^2 \sum_{i=1}^n \mathbf{y}_i^T \mathbf{y}_i$ and $s_{xy} := \sigma_y \sigma_x \sum_{i=1}^n \mathbf{x}_i^T \mathbf{y}_i$. Together, these equations give a closed-form solution for the ML estimator of λ, assuming the measurement error variances σ_x^2 and σ_y^2 are known.

[1] The noise in $\mathbf{x}_i$ does not depend on λ as it is proportional to the average keypoint depth, which is normalized to 1 for the first keyframe.

B. State Prediction and Observation

In this section, we describe the state space, the observation models and the motion model used in the EKF. The state space consists of a total of ten state variables

$$\mathbf{x}_t := (x_t, y_t, z_t, \dot{x}_t, \dot{y}_t, \dot{z}_t, \Phi_t, \Theta_t, \Psi_t, \dot{\Psi}_t)^T \in \mathbb{R}^{10}, \qquad (8)$$

where (x_t, y_t, z_t) denotes the position of the quadrocopter in m and $(\dot{x}_t, \dot{y}_t, \dot{z}_t)$ the velocity in m/s, both in world coordinates. Further, the state contains the roll Φ_t, pitch Θ_t and yaw Ψ_t angle of the drone in deg, as well as the yaw-rotational speed $\dot{\Psi}_t$ in deg/s. In the following, we define for each sensor an observation function $h(\mathbf{x}_t)$ and describe how the respective observation vector $\mathbf{z}_t$ is composed from the sensor readings.

1) **Odometry Observation Model:** The quadrocopter measures its horizontal speed $\hat{v}_{x,t}$ and $\hat{v}_{y,t}$ in its local coordinate system, which we transform into the global frame $\dot{x}_t$ and $\dot{y}_t$. The roll and pitch angles $\hat{\Phi}_t$ and $\hat{\Theta}_t$ measured by the accelerometer are direct observations of Φ_t and Θ_t. To account for yaw-drift and uneven ground, we differentiate the height measurements $\hat{h}_t$ and yaw measurements $\hat{\Psi}_t$ and treat them as observations of the respective velocities. The resulting observation function $h_{\mathrm{I}}(\mathbf{x}_t)$ and measurement vector $\mathbf{z}_{\mathrm{I},t}$ is hence given by

$$h_{\mathrm{I}}(\mathbf{x}_t) := \begin{pmatrix} \dot{x}_t \cos\Psi_t - \dot{y}_t \sin\Psi_t \\ \dot{x}_t \sin\Psi_t + \dot{y}_t \cos\Psi_t \\ \dot{z}_t \\ \Phi_t \\ \Theta_t \\ \dot{\Psi}_t \end{pmatrix} \qquad (9)$$

$$\mathbf{z}_{\mathrm{I},t} := (\hat{v}_{x,t}, \hat{v}_{y,t}, (\hat{h}_t - \hat{h}_{t-1}), \hat{\Phi}_t, \hat{\Theta}_t, (\hat{\Psi}_t - \hat{\Psi}_{t-1}))^T \qquad (10)$$

2) **Visual Observation Model:** When PTAM successfully tracks a video frame, we scale the pose estimate by the current estimate for the scaling factor λ^* and transform it from the coordinate system of the front camera to the coordinate system of the quadrocopter, leading to a direct observation of the quadrocopter's pose given by

$$h_{\mathrm{P}}(\mathbf{x}_t) := (x_t, y_t, z_t, \Phi_t, \Theta_t, \Psi_t)^T \qquad (11)$$

$$\mathbf{z}_{\mathrm{P},t} := f(\mathbf{E}_{\mathcal{DC}} \mathbf{E}_{\mathcal{C},t}) \qquad (12)$$

where $\mathbf{E}_{\mathcal{C},t} \in \mathrm{SE}(3)$ is the estimated camera pose (scaled with λ), $\mathbf{E}_{\mathcal{DC}} \in \mathrm{SE}(3)$ the constant transformation from the camera to the quadrocopter coordinate system, and $f : \mathrm{SE}(3) \to \mathbb{R}^6$ the transformation from an element of SE(3) to our roll-pitch-yaw representation.

3) **Prediction Model:** The prediction model describes how the state vector $\mathbf{x}_t$ evolves from one time step to the next. In particular, we approximate the quadrocopter's horizontal acceleration $\ddot{x}, \ddot{y}$ based on its current state $\mathbf{x}_t$, and estimate its vertical acceleration $\ddot{z}$, yaw-rotational acceleration $\ddot{\Psi}$ and roll/pitch rotational speed $\dot{\Phi}, \dot{\Theta}$ based on the state $\mathbf{x}_t$ and the active control command $\mathbf{u}_t$.

The horizontal acceleration is proportional to the horizontal force acting upon the quadrocopter, which is given by

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} \propto \mathbf{f}_{\mathrm{acc}} - \mathbf{f}_{\mathrm{drag}} \qquad (13)$$

where $\mathbf{f}_{\mathrm{drag}}$ denotes the drag and $\mathbf{f}_{\mathrm{acc}}$ denotes the accelerating force. The drag is approximately proportional to the horizontal velocity of the quadrocopter, while $\mathbf{f}_{\mathrm{acc}}$ depends on the tilt angle. We approximate it by projecting the quadrocopter's z-axis onto the horizontal plane, which leads to

$$\ddot{x}(\mathbf{x}_t) = c_1 \left(\cos\Psi_t \sin\Phi_t \cos\Theta_t - \sin\Psi_t \sin\Theta_t\right) - c_2 \dot{x}_t \qquad (14)$$

$$\ddot{y}(\mathbf{x}_t) = c_1 \left(-\sin\Psi_t \sin\Phi_t \cos\Theta_t - \cos\Psi_t \sin\Theta_t\right) - c_2 \dot{y}_t \qquad (15)$$

We estimated the proportionality coefficients c_1 and c_2 from data collected in a series of test flights. Note that this model assumes that the overall thrust generated by the four rotors is constant. Furthermore, we describe the influence of sent control commands $\mathbf{u}_t = (\bar{\Phi}_t, \bar{\Theta}_t, \bar{\dot{z}}_t, \bar{\dot{\Psi}}_t)$ by a linear model:

$$\dot{\Phi}(\mathbf{x}_t, \mathbf{u}_t) = c_3 \bar{\Phi}_t - c_4 \Phi_t \qquad (16)$$

$$\dot{\Theta}(\mathbf{x}_t, \mathbf{u}_t) = c_3 \bar{\Theta}_t - c_4 \Theta_t \qquad (17)$$

$$\ddot{\Psi}(\mathbf{x}_t, \mathbf{u}_t) = c_5 \bar{\dot{\Psi}}_t - c_6 \dot{\Psi}_t \qquad (18)$$

$$\ddot{z}(\mathbf{x}_t, \mathbf{u}_t) = c_7 \bar{\dot{z}}_t - c_8 \dot{z}_t \qquad (19)$$

Again, we estimated the coefficients $c_3, \ldots, c_8$ from test flight data. The overall state transition function is now given by

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ z_{t+1} \\ \dot{x}_{t+1} \\ \dot{y}_{t+1} \\ \dot{z}_{t+1} \\ \Phi_{t+1} \\ \Theta_{t+1} \\ \Psi_{t+1} \\ \dot{\Psi}_{t+1} \end{pmatrix} \leftarrow \begin{pmatrix} x_t \\ y_t \\ z_t \\ \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \\ \Phi_t \\ \Theta_t \\ \Psi_t \\ \dot{\Psi}_t \end{pmatrix} + \delta_t \begin{pmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \\ \ddot{x}(\mathbf{x}_t) \\ \ddot{y}(\mathbf{x}_t) \\ \ddot{z}(\mathbf{x}_t, \mathbf{u}_t) \\ \dot{\Phi}(\mathbf{x}_t, \mathbf{u}_t) \\ \dot{\Theta}(\mathbf{x}_t, \mathbf{u}_t) \\ \dot{\Psi}_t \\ \ddot{\Psi}(\mathbf{x}_t, \mathbf{u}_t) \end{pmatrix} \qquad (20)$$

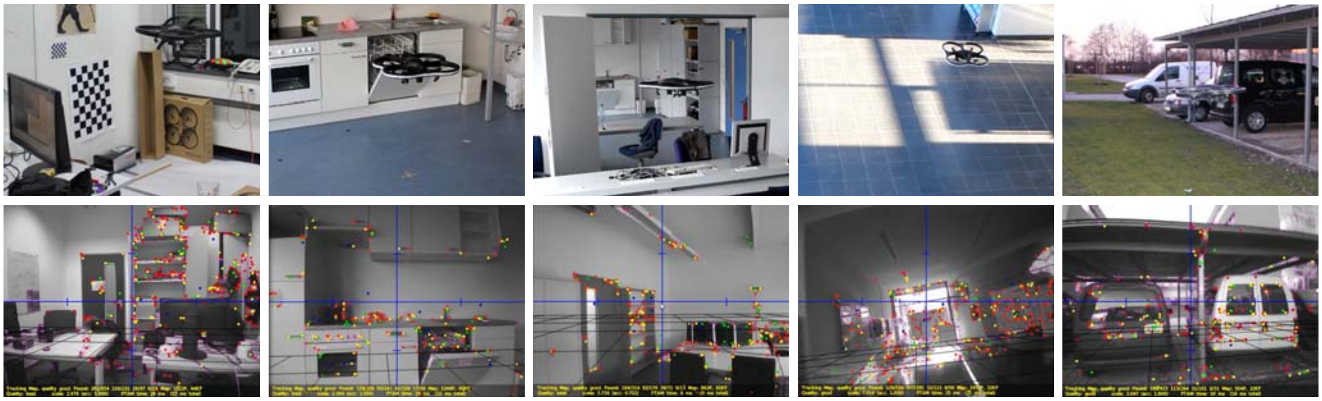
using the model specified in (14) to (19). Note that, due to the many assumptions made, we do not claim the physical correctness of this model. It however performs very well in practice, which is mainly due to its completeness: the behavior of all state parameters and the effect of all control commands is approximated, allowing "blind" prediction, i.e., prediction without observations for a brief period of time (~ 125 ms in practice, see Fig. 3).

V. Experiments and Results

We conducted a series of real-world experiments to analyze the properties of the resulting system. The experiments were conducted in different environments, i.e., both indoor in rooms of varying size and visual appearance as well as outdoor under the influence of sunlight and (slight) wind. A selection of these environments is depicted in Fig. 4.

A. Scale Estimation Accuracy

To analyze the accuracy of the scale estimation method derived in IV-A, we instructed the quadrocopter to fly a fixed figure, while every second a new sample is taken and the scale re-estimated. In the first set of flights, the quadrocopter was commanded to move only vertically, such that the samples mostly consist of altitude measurements. In the second set, the quadrocopter was commanded to fly



small office kitchen large office large indoor area outdoor

Fig. 4. Testing Environments: The top row shows an image of the quadrocopter flying, the bottom row the corresponding image from the quadrocopter's frontal camera. This shows that our system can operate robustly in different, real-world environments.

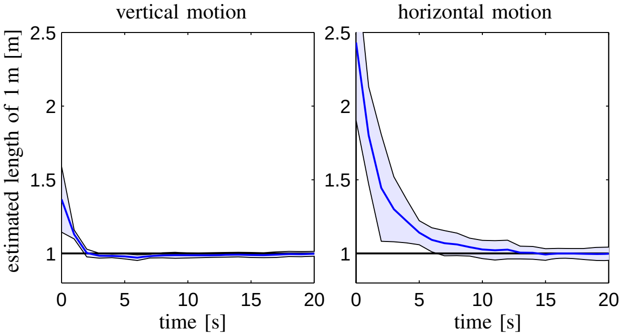


Fig. 5. Scale Estimation Accuracy: The plots show the mean and standard deviation of the the estimation error e, corresponding to the estimated length of 1 m, from horizontal and vertical motion. It can be seen that the scale can be estimated accurately in both cases, it is however more accurate and converges faster if the quadrocopter moves vertically.

a horizontal rectangle, such that primarily the IMU-based velocity information is used. After each flight, we measured the ground truth $\hat{\lambda}$ by manually placing the quadrocopter at two measurement points, and comparing the known distance between these points with the distance measured by the visual SLAM system. Note that due to the initial scale normalization, the values for $\hat{\lambda}$ roughly correspond to the mean feature depth in meters of the first keyframe, which in our experiments ranges from 2 m to 10 m. To provide better comparability, we analyze and visualize the estimation error $e := \frac{\hat{\lambda}^*}{\hat{\lambda}}$, corresponding to the estimated length of 1 m.

Fig. 5 gives the mean error as well as the standard deviation spread over 10 flights. As can be seen, our method quickly and accurately estimates the scale from both types of motion. Due to the superior accuracy of the altimeter compared to the horizontal velocity estimates, the estimate converges faster and is more accurate if the quadrocopter moves vertically, i.e., convergence after 2 s versus 15 s, and to a final accuracy ±1.7 % versus ±5 %. Note that in practice, we allow for (and recommend) arbitrary motions during scale estimation so that information from both sensor modalities can be used to improve convergence. Large, sudden changes in measured relative height can be attributed to uneven ground, and removed automatically from the data set.

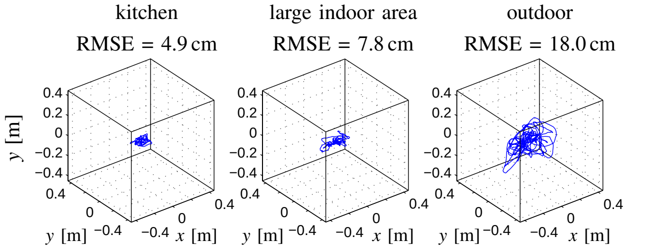


Fig. 6. Flight Stability: Path taken and RMSE of the quadrocopter when instructed to hold a target position for 60 s, in three of the environments depicted in Fig. 4. It can be seen that the quadrocopter can hold a position very accurately, even when perturbed by wind (right).

B. Positioning Accuracy

In this section, we evaluate the performance of the complete system in terms of position control. We instructed the quadrocopter to hold a target position over 60 s in different environments and measure the root mean square error (RMSE). The results are given in Fig. 6: the measured RMSE lies between 4.9 cm (indoor) and 18.0 cm (outdoor).

C. Drift Elimination

To verify that the incorporation of a visual SLAM system eliminates odometry drift, we compare the estimated trajectory with and without the visual SLAM system. Fig. 7 shows the resulting paths, both for flying a fixed figure (left) and for holding a target position while the quadrocopter is being pushed away (right). Both flights took approximately 35 s, and the quadrocopter landed no more than 15 cm away from its takeoff position. In contrast, the raw odometry accumulated an error of 2.1 m for the fixed figure and 6 m when being pushed away - which is largely due to the relative lack of texture on the floor. This experiment demonstrates that the visual SLAM system efficiently eliminates pose drift during maneuvering.

D. Figure Flying

Based on the accurate pose estimation and control, we implemented a simple scripting language for flying prespecified maneuvers. Available commands in this scripting language include take-off, landing, automatic initialization

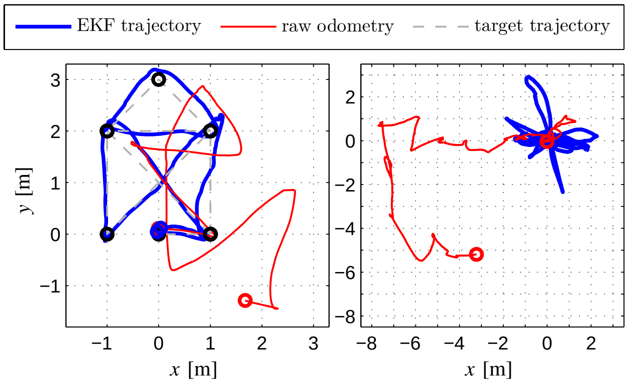


Fig. 7. Elimination of Odometry Drift: Horizontal path taken by the quadrocopter as estimated by the EKF compared to the raw odometry (i.e., the integrated velocity estimates). Left: when flying a figure; right: when being pushed away repeatedly from its target position. The odometry drift is clearly visible, in particular when the quadrocopter is being pushed away. When incorporating visual pose estimates, it is eliminated completely.

(take-off + PTAM map initialization) and approaching a waypoint, both in absolute coordinates (with the world origin at the take-off location) as well as relative to the current location. Various parameters can be set during the flight, for example to re-define the world origin, limit flight speed, and the parameters for approaching a waypoint. A waypoint has been "reached" after the quadrocopter reaches and remains within a certain distance (default: 0.5 m) for a certain time (default: 2.0 s). With this scripting language, we were able to let the quadrocopter autonomously complete a large variety of different figures including take-off and map initialization, for example a rectangle and the "Haus vom Nikolaus" as depicted in Fig. 1, both vertically and horizontally. Demonstrations of these flight patterns are also shown in the video accompanying this paper.

VI. Conclusion

In this paper, we presented a visual navigation system for autonomous figure flying. Our system enables the quadrocopter to visually navigate in unstructured, GPS-denied environments and does not require artificial landmarks nor prior knowledge about the environment. We tested our system in a set of extensive experiments in different real-world indoor and outdoor environments and with different flight patterns. We found in these experiments, that our system achieves an average positioning accuracy of 4.9 cm (indoor) to 18.0 cm (outdoor) and can robustly deal with communication delays of up to 400 ms. With these experiments, we demonstrated that accurate, robust and drift-free visual figure flights are possible.

We plan to release our software as open-source in the near future.

References

[1] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[2] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips." in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[3] M. Müller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling," in *Proc. IEEE Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[4] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," in *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, 2011.

[5] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2009.

[6] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[7] 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 *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[8] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," in *Proc. SPIE Unmanned Systems Technology XI*, 2009.

[9] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Proc. IEEE International Symposium of Robotics Research (ISRR)*, 2011.

[10] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[11] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. IEEE Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[12] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proceedings of the Intl. Symposium on Experimental Robotics*, Dec 2010.

[13] D. Eberli, D. Scaramuzza, S. Weiss, and R. Siegwart, "Vision based position control for MAVs using one single circular landmark," *Journal of Intelligent and Robotic Systems*, vol. 61, pp. 495–512, 2011.

[14] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "AR-drone as a platform for robotic research and education," in *Proc. Research and Education in Robotics: EUROBOT 2011*, 2011.

[15] "Ascending technologies," 2012. [Online]: http://www.asctec.de/

[16] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[17] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "MAV navigation through indoor corridors using optical flow," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[18] "Parrot AR.Drone," 2012. [Online]: http://ardrone.parrot.com/

[19] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[20] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart, "Fusion of IMU and vision for absolute scale estimation in monocular SLAM," *Journal of Intelligent Robotic Systems*, vol. 61, pp. 287 – 299, 2010.

[21] C. Bills, J. Chen, and A. Saxena, "Autonomous MAV flight in indoor environments using single image perspective cues," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[22] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "AR-drone as a platform for robotic research and education," in *Proc. Communications in Computer and Information Science (CCIS)*, 2011.

[23] W. S. Ng and E. Sharlin, "Collocated interaction with flying robots," in *Proc. IEEE Intl. Symposium on Robot and Human Interactive Communication*, 2011.