Monocular-SLAM–Based Navigation for Autonomous Micro Helicopters in GPS-Denied Environments

Stephan Weiss
Autonomous Systems Lab, ETH Zurich, Zurich 8092, Switzerland
e-mail: stephan.weiss@mavt.ethz.ch
Davide Scaramuzza
GRASP Lab, University of Pennsylvania, Philadelphia, Pennsylvania 19704
e-mail: davide.scaramuzza@ieee.org
Roland Siegwart
Autonomous Systems Lab, ETH Zurich, Zurich 8092, Switzerland
e-mail: r.siegwart@ieee.org

Received 26 November 2010; accepted 24 June 2011

Autonomous micro aerial vehicles (MAVs) will soon play a major role in tasks such as search and rescue, environment monitoring, surveillance, and inspection. They allow us to easily access environments to which no humans or other vehicles can get access. This reduces the risk for both the people and the environment. For the above applications, it is, however, a requirement that the vehicle is able to navigate without using GPS, or without relying on a preexisting map, or without specific assumptions about the environment. This will allow operations in unstructured, unknown, and GPS-denied environments. We present a novel solution for the task of autonomous navigation of a micro helicopter through a completely unknown environment by using solely a single camera and inertial sensors onboard. Many existing solutions suffer from the problem of drift in the xy plane or from the dependency on a clean GPS signal. The novelty in the here-presented approach is to use a monocular simultaneous localization and mapping (SLAM) framework to stabilize the vehicle in six degrees of freedom. This way, we overcome the problem of both the drift and the GPS dependency. The pose estimated by the visual SLAM algorithm is used in a linear optimal controller that allows us to perform all basic maneuvers such as hovering, set point and trajectory following, vertical takeoff, and landing. All calculations including SLAM and controller are running in real time and online while the helicopter is flying. No offline processing or preprocessing is done. We show real experiments that demonstrate that the vehicle can fly autonomously in an unknown and unstructured environment. To the best of our knowledge, the here-presented work describes the first aerial vehicle that uses onboard monocular vision as a main sensor to navigate through an unknown GPS-denied environment and independently of any external artificial aids.

1. INTRODUCTION

Small-scale autonomous aerial vehicles will play a major role in the near future. Micro helicopters—and especially quadrocopters—will be of particular interest because of their great agility and ability to perform fast maneuvers (Mellinger, Michael, & Kumar, 2010). They will accomplish tasks in areas such as surveillance, search and rescue, monitoring, mapping, and other areas in which an agile ground-decoupled system is needed. The research in autonomous micro aerial vehicles (MAVs) is advancing fast, but even though a lot of progress has been achieved in this topic during the past few years, we still strive to find autonomous systems in unknown, cluttered, and GPS-denied environments. Only after solving this issue can high-level tasks such as autonomous exploration, swarm navigation, and large trajectory planning be tackled (Belloni, Feroli, Ficola, Pagnottelli, & Valigi, 2007; Birk, Wiggerich, Bulow, Pfingsthorn, & Schwertfeger, 2011; Dhaliwal & Ramirez-Serrano, 2009; Goodrich et al., 2007; Maza, Caballero, Capitán, de Dios, & Ollero, 2011; Merino, Caballero, de Dios, Ferruz, & Ollero, 2006).

A key problem on an airborne vehicle is the stabilization and control in six degrees of freedom (DOF), that is, attitude and position control, Today's systems handle well the attitude control. Without a position control, they are, however, prone to a position drift over time. Stable flights and navigation with GPS are well explored and work out of the box (AscTec, N.D.). However, GPS is not a reliable service as its availability can be limited by urban canyons and it is completely unavailable in indoor environments. The alternative of using laser range finders is not optimal because these sensors have a restricted perception distance and are still heavy for MAVs.

Multimedia files may be found in the online version of this article.

Considering what we have mentioned so far, in order to be independent of the (quality of the) GPS signal, a viable solution is to navigate using a vision-based system. Stable flights and navigation with external (i.e., offboard) cameras—such as the motion capture systems—are well explored (How, Bethke, Frank, Dale, & Vian, 2008; Michael, Fink, & Kumar, 2010; Valenti, Bethke, Fiore, & How, 2006). However, they have numerous disadvantages: they are limited to small environments, the operation of the helicopters is constrained within the cameras' field of view, and they require manual installation and calibration of the cameras, and thus they are limited to environments that are physically accessible by humans. The alternative and only viable solution to make the vehicle free to explore places that are not accessible by humans is to install the sensors onboard. In this work we focus on vision sensors because laser-based systems are still much heavier and consume much more power. In the past few years, development on digital cameras has made them fulfill two important requirements for MAVs, low weight and low power consumption. In addition, the quality of the cameras—in terms, for instance, of contrast, resolution, frame rate, high dynamic range, global shutter—increased, while their price dropped remarkably. The main advantage of an onboard camera is that it provides very rich and distinctive information. This is definitely useful for structure and motion estimation and relocalization. Monocular or stereo cameras offer both a good solution. However, we must remember that a stereo camera loses its effectiveness if the distance to the scene is much larger than the baseline. In this case, the range data become inaccurate and the utility of the stereo reduces to that of the monocular camera.

In this paper, we show that full navigation of a micro helicopter becomes possible using only a single camera as exteroceptive sensor and an inertial measurement unit (IMU) for proprioceptive measurements. We show autonomous navigation in GPS-denied and completely unknown environments. The here-presented approach uses the camera to build a sparse three-dimensional (3D) map of the environment, localize with respect to the map, and finally navigate through it. The entire process runs in real time during the flight. The difference of the estimated camera position in the 3D map and a corresponding setpoint is used as error in a linear quadratic Gaussian/loop transform recovery (LQG/LTR)-based controller. We show that minimizing this error allows all basic flight maneuvers such as hovering, set point and trajectory following, vertical takeoff, and landing. We present many experimental results and show that the combination of these basic tasks leads us to fully autonomous flight, from takeoff to landing. To the best of our knowledge, this work describes the first micro helicopter platform that uses monocular simultaneous localization and mapping (SLAM) to navigate through an unexplored GPS-denied environment and independently of any artificial aid such as external beacons, offboard cameras, and markers.

The outline of the paper is as follows. The related work is described in Section 2. We introduce the notation in Section 3. In Section 4, we summarize the SLAM algorithm that has been used and explain the reason for this choice for our approach. In Section 5, we describe the modeling of the system and the parameter identification. After that, we discuss the controller design in Section 6 and analyze the entire structure of our approach in Section 7. In Section 8, we determine the visual-SLAM algorithm's accuracy and focus on our technique for local navigation. Finally, we discuss the achieved results and conclude the paper in Section 9.

2. RELATED WORK

Because MAVs are in general highly unstable and nonlinear systems, a clever combination of sensor equipment and controller must be designed. Most of the approaches model the MAV as two connected ideal subsystems and use a cascaded control structure: one controller for the attitude of the MAV (3D orientation of the helicopter) and one superposed controller for its 3D position. In general, the attitude controller has to be faster than the position controller because of the vehicle's high dynamics. Often, a simple proportional–derivative (PD)-controller is enough for this purpose; however, other techniques can also be applied for its design (Cai, Chen, & Lee, 2008; Castillo, Lozano, & Dzul, 2004; Peng et al., 2007). Bouabdallah and Siegwart (2005) analyzed the application of two different control techniques—"sliding-mode" and "backstepping"—and showed that the latter has particularly good stabilizing qualities. In our case, we use the onboard attitude controller provided by Ascending Technologies (AscTec, N.D.). It is basically a PD controller running at 1 kHz.

The use of accelerometers, gyros, and compasses can lead to strong attitude controllers and enables the stabilization of the vehicle's attitude. Although it is possible to hold the flying platform in a hovering state, there is no possibility to perceive and correct for any drift caused by accumulated errors. To tackle this issue, exteroceptive sensors (e.g., laser scanners, cameras, GPS, pressure sensors, or ultrasonic sensors) able to measure the vehicle position are unavoidable. The most common approach is to mount a DGPS receiver on the MAV. By using a so-called inertial/GPS approach, in which data from an IMU and GPS data are fused together, the position can be estimated up to a precision of some decimeters. Thus, the MAV can be fully stabilized and controlled (Abdelkrim, Aouf, Tsourdos, & White, 2008; Yun, Peng, & Chen, 2007). The two drawbacks of this approach are the necessity to receive a clean GPS signal and the lack of precision of the position estimate. Furthermore, GPS is not a reliable service; its availability can be limited by urban canyons, and it is completely unavailable in indoor environments. Nevertheless, most approaches for autonomous MAVs have been developed using GPS, sometimes in combination with onboard cameras and an IMU. For instance, Ludington, Johnson,

and Vachtsevanos (2006) used a Kalman filter to fuse GPS, vision, and IMU data. Templeton, Shim, Geyer, and Sastry (2007) used a GPS-based flight control system for navigation and vision to estimate the 3D structure of the terrain in order to find adequate landing sites.

Alternatively to GPS, successful results have recently been achieved using laser range finders (Achtelik, Bachrach, He, Prentice, & Roy, 2009; Bachrach, He, & Roy, 2009a, 2009b). Bachrach et al. (2009a, 2009b) used an Hokuyo laser scanner and two-dimensional (2D) SLAM for autonomous navigation in a maze. With their platform, they won the international competition of MAVs (IMAV), which consisted of entering and exiting from a maze. In contrast to cameras, laser range finders have the advantage of working in texture-less environments. Although very appealing, the use of range finders is not optimal because these sensors have a restricted perception distance and limited field of view (typically only in a plane) and are still heavy for MAVs. We note that both the power consumption of the sensor itself and the energy to lift its weight have to be taken into account for the system's energy budget.

As mentioned previously, and given a limited weight and energy budget, the most viable solution for GPS-denied environments is to use vision. The most simple way is to install a number of external cameras with known location and to have them track the vehicle (Altug, Ostrowski, & Mahony, 2002; Klose et al., 2010; Park et al., 2005). In the past few years, excellent results in this endeavor have been achieved using the motion capture system from Vicon[1]: a system of high-resolution, external cameras that can track the six-DOF pose of one or more helicopters with submillimeter accuracy (How, Bethxe, Frank, Dale, & Vian, 2008; Michael, Mellinger, Lindsey, & Kumar, 2010; Valenti et al., 2006). This is made possible by the use of retroreflective markers that are rigidly attached to the vehicle's body and that can easily be tracked by the Vicon's cameras even when all but one camera are occluded. Using the software from Vicon, tracking of quadrotors is rarely lost, even during extreme situations such as fast maneuvers (speed of 3.5 m/s, acceleration of 15 m/s^2) (Michael, Mellinger, et al., 2010). Aggressive maneuvers such as multiple flips of the MAV (Lupashin, Schöllig, Sherback, & D'Andrea, 2010), docking to an oblique wall (Mellinger, Michael, et al., 2010), or cooperative grasping of objects (Mellinger, Shomin, Michael, & Kumar, 2010; Michael, Fink, & Kumar, 2010) were achieved in 2010 using the Vicon system. Typically, in these installations the number of Vicon cameras for MAV test beds ranges between 1 and 16, the resolution can be up to 16 megapixels, and the frame rate can reach 375 Hz. Motion capture systems are very efficient and robust for testing purposes and can be used as ground truth reference to evaluate other approaches. However, they are obviously not suitable for large environments and for missions in which the installation of an appropriate infrastructure for external cameras is not feasible. Additionally, they require an accurate calibration of the camera system. Finally, we also point out the difference between onboard cameras and external (i.e., offboard) cameras: when using external cameras (such as the Vicon), it is not the helicopter that is autonomous but rather the system comprising the external cameras plus the helicopter. It is only in the case of onboard cameras that the helicopter can be considered truly autonomous.

Most vision-based works developed so far concentrated on specific, individual tasks only such as takeoff, landing, hovering, or obstacle avoidance. For instance, Saripalli, Montgomery, and Sukhatme (2002) and Mejias, Usher, Roberts, and Corke (2006) focused on autonomous landing using only vision or vision and IMU, respectively. In particular, Mejias et al. (2006) used vision to identify a good landing spot and reach it as fast and safely as possible. Most vision-based controllers for hovering purposes only have been implemented by means of optical flow (Herisse, Russotto, Hamel, & Mahony, 2008; Zufferey & Floreano, 2006). Herisse et al. (2008) used an optical flow–based PI controller to stabilize a hovering MAV. They also implemented an automatic landing routine by analyzing the divergent optical flow. However, optical flow–based hovering is affected by drift because it computes the relative displacement of the features between the last two frames and not their absolute motion, for example, with respect to the initial frame. This also holds for visual odometry–based implementations, in which the pose is estimated by considering the feature displacements between two consecutive images (Fowers et al., 2007). Optical flow, however, turns out to be extremely useful for obstacle avoidance and wall or terrain following. Based on optical flow, some biologically inspired control algorithms have been developed in this endeavor, as reported in Garratt and Chahl (2008), Hrabar, Sukhatme, Corke, Usher, and Roberts (2005), Ruffier and Franceschini (2004), and Zufferey and Floreano (2006). For example, a wall-collision avoidance was developed by Hrabar et al. (2005), who implemented a strategy to navigate in urban canyons. They used optical flow from two cameras placed on both sides of the vehicle. In addition, they used a front-looking stereo camera to avoid oncoming obstacles. In our previous work (Zingg, Scaramuzza, Weiss, & Siegwart, 2010), we also used optical flow to control a quadrocopter to remain in the center of a corridor. Garratt and Chahl (2008) used optical flow to perform terrain following with a coaxial helicopter.

Because optical flow alone is not a good choice for fully autonomous helicopters, the only viable solution is to extract and track distinctive and reliable features that can easily be matched across multiple frames. In general, two approaches are possible when using onboard cameras: the first one is by tracking a known, fixed object (such as artificial markers, or user-specified points), which

[1]http://www.vicon.com.

implicitly solves the matching and relocalization problem because the markers are already known and their relative 3D position too; the second approach is by extracting distinctive natural features—whose 3D position is a priori not known—and use them for both motion and structure estimation and relocalization (i.e., visual SLAM). Many different approaches have been developed within the first category (Hamel, Mahony, & Chriette, 2002; Proctor & Johnson, 2004). Hamel et al. (2002) implemented a visual trajectory tracking method to control a MAV with an onboard camera observing n fixed points. A similar approach was developed by Cheviron, Hamel, Mahony, and Baldwin (2007), who additionally fused the vision with IMU data. Another possibility is to have the MAV tracking a leading MAV and maintain fixed relative position and orientation. Chen and Dawson (2006) implemented this by tracking some coplanar points on the leading vehicle and using homography to estimate the relative pose. In contrast, Wenzel, Masselli, and Zell (2010) used four light sources on a ground robot (a Pioneer) and homography to perform autonomous takeoff, tracking, and landing on the moving ground robot.

To the best of our knowledge, very little work has been done within the second category, which is using a full visual SLAM algorithm for controlling the helicopter. These works are described in Artieda et al. (2008) and Ahrens, Levine, Andrews, and How (2009). The advantage of visual SLAM compared to all the approaches mentioned so far is that it allows the MAV to cancel the motion drift when returning to an already-visited location. The work of Artieda et al. (2008) describes the implementation of a visual SLAM based on an extended Kalman filter (EKF) for UAVs, but their visual SLAM is used only for mapping and not for controlling the helicopter. Therefore, the work that is probably closer to ours is that from Ahrens et al. (2009). On the basis of the monocular SLAM algorithm of Davison, Reid, Molton, and Strasse (2007), they built a localization and mapping framework that provides an almost drift-free pose estimation. With that, they implemented a position controller and obstacle avoidance. However, due to the simplification they used in their feature-tracking algorithm, a nonnegligible drift persists. In addition, they used the Vicon motion capture system to control the aerial vehicle. Therefore, the output of the SLAM-based localization system was not used for vehicle stabilization. Another problem is represented by the choice of an EKF-based visual SLAM algorithm that is particularly sensitive to outliers, abrupt motion, and occlusions and that does not perform relocalization if the helicopter is "kidnapped." Another major difference is that they focus only on navigation indoors and do not cover autonomous takeoff and landing, which, in contrast, is tackled in the present work.

In this paper, we present an approach based on the visual SLAM algorithm of Klein and Murray (2007), which—as recently shown by Strasdat, Montiel, and Davison (2010)—is more robust than filter-based visual SLAM implementations and therefore more suitable for MAV applications. It enables the airborne vehicle to autonomously determine its location and consequently stabilize itself. In contrast to other approaches, we do not require any a priori information on the environment, nor do we need external aids such as an external tracking system in order to obtain a position and attitude control for the MAV. The controller is based on a cascaded structure and is designed by means of a discrete LQG/LTR procedure applied on a simplified MAV model. This enables us to handle the considerable time delay that comes from the image processing and from the SLAM algorithm.

For the experimental tests, a downward-looking camera is mounted on a quadrocopter from Ascending Technologies.[2] The images are fed to the visual SLAM algorithm running onboard an Atom computer. Based on the position estimate from the SLAM, the control inputs are computed and then used to control the quadrotor. To the best of our knowledge, this is the first implementation of a closed-loop, vision-based MAV navigation system for unknown GPS-denied environments, which runs fully onboard and without the aid of external beacons, offboard cameras, and markers. We remark that with this system we ranked second in the European MAV competition (EMAV'09) in September 2009 in Delft, The Netherlands. Our helicopter was the only autonomous vehicle using vision (no laser range finder or external beacon was used). The task consisted of taking off, approaching a small apartment, and passing through its window as shown in one of the videos attached to this paper.

3. NOTATION

To facilitate the following considerations we will introduce some notation. We always use boldface for vectors.

Common notations:

${}_A\boldsymbol{v}$	Vector $\boldsymbol{v}$ expressed in the A coordinate system
R_{AB}	Rotation matrix from coordinate system B to coordinate system A

Coordinate systems:

I	Inertial coordinate system, chosen so that the gravity lies along the z axis
M	Coordinate system of the map of the SLAM algorithm
C	Coordinate system of the camera frame
H	Coordinate system of the helicopter

Vectors and scalars:

$\boldsymbol{r}$	Position vector of the helicopter
$\boldsymbol{T}$	Thrust vector of the helicopter (always lies on the z axis of the H coordinate frame).

[2] http://www.asctec.de.

T	The absolute value of the thrust vector $\boldsymbol{T}$
φ	Roll angle of the helicopter, rotation around the x axis of the I coordinate system
θ	Pitch angle of the helicopter, rotation around the y axis of the I coordinate system
ψ	Yaw angle of the helicopter, rotation around the z axis of the I coordinate system
ω	Rotational speed around the z axis of the I coordinate system

Constant parameters:

$\boldsymbol{F}_G$	Gravitational force
g	Gravitational acceleration
m	Mass of the helicopter

Please note that we use the Tait–Bryan convention for the Euler decomposition of the rotation matrix R_{HI} into the three angles φ, θ, and ψ. If the angles represent rotations between two coordinates frames other than I and H, we specify them in the index, e.g., ψ_{CM} represents the rotation around the z axis from the map coordinate frame to the camera coordinate frame. All coordinate frames have the same invariant origin.

Estimated values are denoted by an additional tilde (e.g., ${}_M\tilde{\boldsymbol{r}}$). Reference values are denoted with a star (e.g., T^*).

4. VISUAL SLAM-BASED LOCALIZATION

4.1. Description of the Visual SLAM Algorithm

The here-presented approach uses the visual SLAM algorithm of Klein and Murray (2007) in order to localize the MAV with a single camera (see Figure 1). In summary, Klein and Murray split the SLAM task into two separate threads: the tracking thread and the mapping thread. The tracking thread is responsible for the tracking of salient features in the camera image, i.e., it compares the extracted point features with the stored map and thereby attempts to determine the position of the camera. This is done with the following steps: first, a simple motion model is applied to predict the new pose of the camera. Then the stored map points are projected into the camera frame, and corresponding features [FAST corners (Rosten & Drummond, 2005)] are searched. This is also often referred to as the data association. When this is done, the algorithm refines the orientation and position of the camera such that the total error between the observed point features and the projection of the map points into the actual frame is minimized. The mapping thread uses a subset of all camera images—also called key frames—to build a 3D-point map of the surroundings. The key frames are selected using some heuristic criteria. After adding a new key frame, a batch optimization is applied to refine both the map points and the key frame poses. This attempts to minimize the total error between the reprojected map points and the corresponding observations in the key frames. In the computer vision community, this procedure is also referred to as bundle adjustment.

There are several important differences between the key frame–SLAM algorithm considered here and the standard filter-based approach [e.g., Davison et al. (2007)]. First, Klein and Murray's algorithm does not use an EKF-based state estimation and does not consider any uncertainties, either for the pose of the camera or for the location of the features. This saves a lot of computational effort. Considering the uncertainty of the state could ease the data association process. The lack of modeling uncertainties, however, is compensated by the use of a large number of features and the global batch optimization. Therefore, despite using a fixed area for feature matches, the algorithm is still able to track efficiently the point features and to close loops. This makes the algorithm extremely fast and reliable and the map very accurate. As demonstrated in a recent paper by Strasdat et al. (2010), key frame SLAM outperforms filter-based SLAM.

4.2. Analysis of the SLAM Algorithm

The main advantage of the thread splitting lies in that both the mapping and the tracking thread can run at different frequencies. Thus, the mapping thread is able to apply a much more powerful and time-consuming algorithm to build its map. Simultaneously, the tracking thread can estimate the camera pose at a higher frequency. This does strongly improve the performance. Compared to frame-by-frame SLAM (such as EKF SLAM), the algorithm of Klein and Murray (2007) saves a lot of computational effort in that it does not process every single image. Particularly when using a camera with a wide field of view, consecutive images often contain a lot of redundant information. In addition, for example, when the camera is moving very slowly or if it stays at the same position, the mapping thread rarely evaluates the images and, thus, requires only very little computational power. This is the main reason that we chose this monocular SLAM algorithm.

The fact that we mounted the camera down looking increases the overlapping image portion of neighboring key frames, so that we can even further loosen the heuristics for adding key frames to the map. In addition, once the MAV has explored a certain region, no new key frames will be added within the region boundaries. The computation time thus remains constant. On the other hand, when exploring new areas the global bundle adjustment can be very expensive, limiting the number of key frames to a few hundred on our platform (around 50–100, depending on the key frame heuristics).

Another strength of this monocular SLAM algorithm is its robustness against partial camera occlusion. If a sufficient part (around 50%) of the point features can still be tracked, the pose estimate is accurate enough to sustain stable MAV control. In addition, the algorithm will avoid

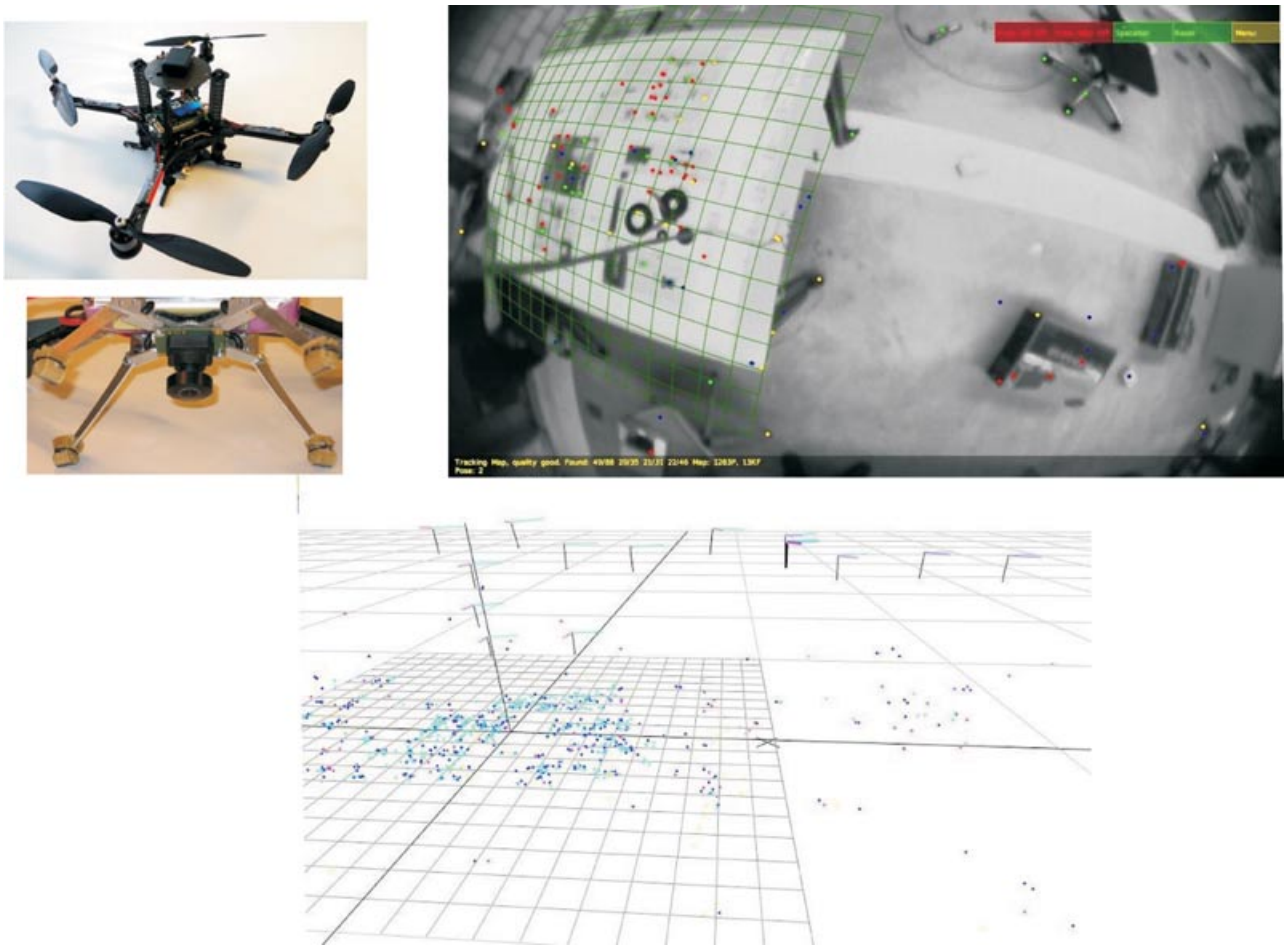


Figure 1. The top-left picture depicts our vehicle (the Pelican) from Ascending Technologies; beneath it, the onboard-mounted camera. The top-right picture is a screenshot of Klein and Murray's visual SLAM algorithm. The tracking of the FAST corners can be observed. This is used for the localization of the camera. In the bottom picture, the 3D map built by the mapping thread is shown. The three-axis coordinate frames represent the location where new key frames were added.

adding any key frames in such a situation so as not to corrupt the map.

An intricate hurdle when using a monocular camera is the lack of any depth information. Because of this, the algorithm must initialize new points based on the observations from more than one key frame. This could motivate the use of a stereo camera. However, for a stereo camera to bring any further advantage, the observed scene must be within some range of the stereo camera; otherwise a single camera will yield the same utility. Closely linked to this problem is the unobservability of the map scale. To tackle this, we initially set the map scale by hand. An EKF is then used to maintain a consistent scale estimate by opportunely fusing IMU data and camera poses. This is described in detail in our previous work (Nutzi, Weiss, Scaramuzza, & Siegwart, 2010).

4.3. Adaptations to the SLAM Algorithm

We adapt some parameters of Klein and Murray's (2007) visual SLAM algorithm to increase its performance within our framework. First, we use a more conservative key frame–selection heuristics in order to decrease the number of key frames added during the map expansion. Thereby, we are able to map much larger areas before reaching the computational limit. Additionally, we reduce the number of points being tracked by the tracking thread from 1,000 to 200. This again increases the maximal map size and the frame rate, while keeping the accurate tracking quality and the robustness against occlusion. Last, in order to increase the tracking quality when hovering over very uneven environments, we increase the depth range at which new map points are added.

A point to note for the later controller design is the speed of the visual SLAM algorithm. On our setup, we can observe very varying time steps between consecutive outputs of the pose estimates. Depending on the actual state of the mapping thread, 10 to 40 pose estimations per second are obtained. This yields us a Nyquist frequency of around 5 Hz (worst case).

To increase the autonomy of the system, we write a routine that can store and load maps. With this, we are able to overcome the initialization of the map during which no position estimate is available. This provides the possibility to start the helicopter from a small known patch and to skip the initialization process. Later the loaded map of the patch can be expanded by exploring the environment.

5. MODELING AND PARAMETER IDENTIFICATION

The MAV platform used in our experiments is the Pelican quadrotor from Ascending Technologies (AscTec, N.D.) (Figure 1). The sensors on the platform, which we got out of the box, include three gyroscopes, a 3D compass, and an accelerometer. At this point, it is important to mention that an efficient attitude controller is implemented on the onboard microcontroller of the helicopter. This permits us to focus on the design of a controller for the stabilization of the x–y–z-position coordinates and the yaw angle.

We produce a model of the system and use the reference values of the attitude controller as the model's control inputs. The output is the pose of the camera, i.e., the 3D position and orientation of the camera in the coordinate frame of the stored map. Thus, the dynamics of the internal attitude controller have to be included in the model.

The attitude controller is designed to control the two tilt angles φ, θ, the angular velocity around the vertical axis ω, and the total thrust T of the helicopter. Therefore, the corresponding reference values of the attitude controller (denoted by φ^*, θ^*, T^*, and ω^*) are the inputs to the model, and the outputs are the estimation of the helicopter position ${}_M\tilde{\boldsymbol{r}}$ and the yaw angle $\tilde{\psi}_{CM}$ computed by decomposing the rotation matrix $\tilde{R}_{CM} \cdot {}_M\tilde{\boldsymbol{r}}$ and $\tilde{R}_{CM}$ are obtained through the SLAM algorithm and can also be transformed to the I coordinate frame.

The helicopter is modeled as a simple point mass on which we apply the principle of linear momentum (Newton's second law). The forces that act on the helicopter are reduced to the thrust force $\boldsymbol{T}$ aligned with the z axis of the helicopter and the gravitational force $\boldsymbol{F}_G$ pointing toward the positive z axis of the inertial coordinate system (see Figure 2). We can now project $\boldsymbol{F}_G$ onto the x–y–z axes and apply the principle of linear momentum onto the three directions of the inertial coordinate frame. This yields the following matrix equation:

$$ {}_I\ddot{\boldsymbol{r}} = R_{IM} \cdot {}_M\ddot{\boldsymbol{r}} = \frac{1}{m} R_{HI}^{-1}(\varphi, \theta, \psi) \begin{pmatrix} 0 \\ 0 \\ -T \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}. \quad (1) $$

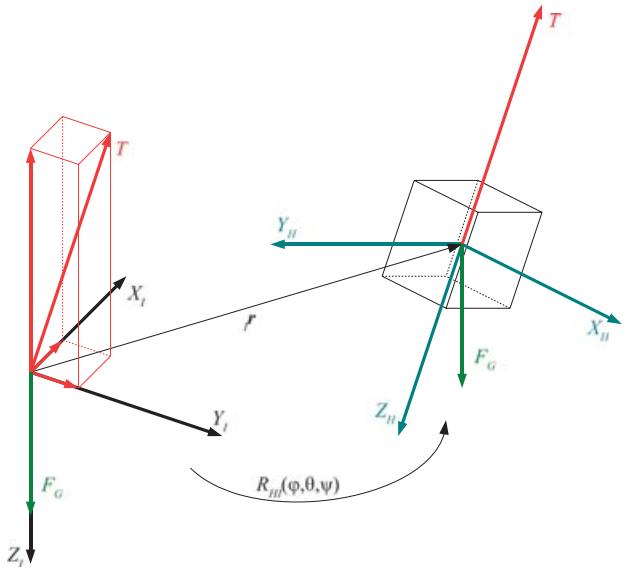


Figure 2. Point mass model of the helicopter. The inertial coordinate frame and the helicopter coordinate frame are illustrated. Using the rotation matrix R_{HI}, the thrust force $\boldsymbol{T}$ and the gravitational force $\boldsymbol{F}_G$ can be projected onto the three axes of the inertial coordinate frame. Subsequently the principle of linear momentum can be applied.

Now the state ${}_M\boldsymbol{r}$ can be computed given the three angles φ, θ, ψ and the thrust value T. For the yaw angle ψ and the thrust value T, we assume the following simple relations:

$$ \dot{\psi} = \omega^*, \quad T = T^*. \quad (2) $$

Note that it is important that the z axis of the inertial frame points toward the center of gravity. Otherwise, the gravity vector is added in the wrong direction. At the moment this is adjusted manually.

As the attitude controller dynamics from the inputs φ^* and θ^* to the angles φ and θ is quite fast, we model them as two separated second-order systems with the following transfer function:

$$ T(s) = \frac{\omega^2}{s^2 + 2 \cdot d \cdot \omega \cdot s + \omega^2}. \quad (3) $$

We then identify the parameters d and ω on the plant by analyzing the step response (see Figure 3). This yields values of 15.92 rad/s for ω and 1.22 for d.

For the subsequent controller design, we need to estimate the time delay T_d in the control loop. This was done by observing the same step response as before. We assume that the time delay is mainly caused by the SLAM algorithm (maximum 80.6 ms), so that its value is the same for all outputs.

The block diagram of the entire system is represented in Figure 4. Note that for the sake of simplicity, we do not model disturbances or noise.

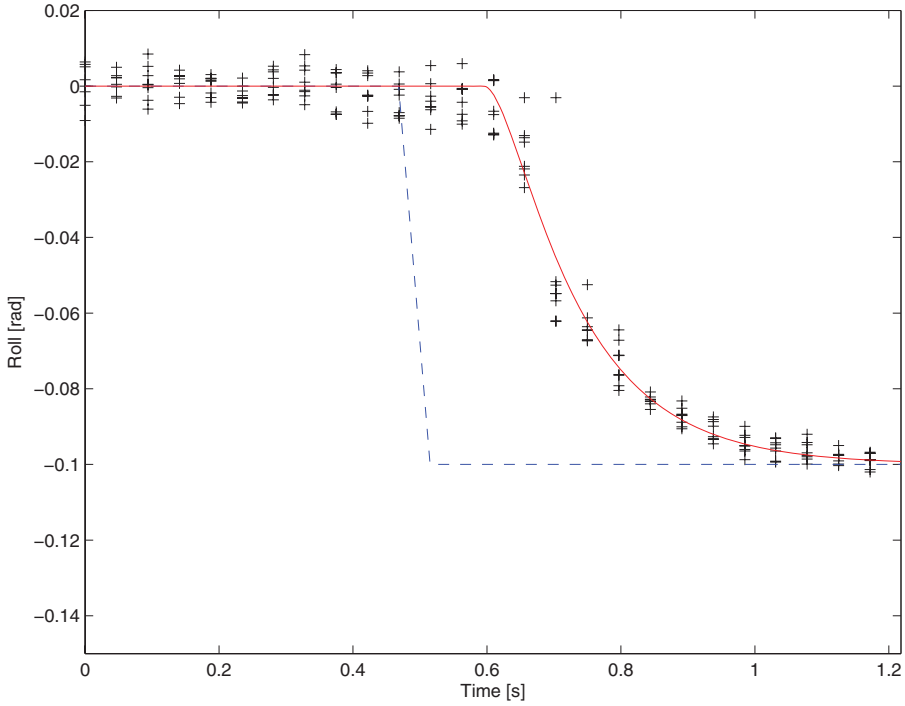


Figure 3. Measured step response on the pitch input of the quadrotor device. The system is modeled as a second-order system with time delay. This yields a critical frequency ω of 15.92 rad/s, a damping d of 1.22, and a time delay T_d of 80.6 ms.

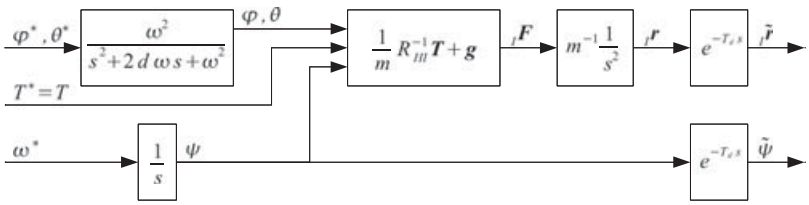


Figure 4. Model of the entire system with the roll angle φ^*, the pitch angle θ^*, the total thrust T^*, and the yaw rate ω^* as inputs. The outputs are the position of the helicopter ${}_I\tilde{\mathbf{r}}$ and the yaw angle $\tilde{\psi}$. The attitude controller dynamics and the time delay are included in the model. Note that external disturbances and noise are not modeled.

6. CONTROLLER DESIGN

As already mentioned, the design of the position controller is based on a plant model in which the dynamics of the attitude controller are included. Unfortunately, the control inputs introduce strong nonlinearities into the system, as can be seen in Eq. (1). Using the Tait–Bryan convention, the following equations for the force acting on the helicopter are derived:

$$\begin{pmatrix} {}_I F_x \\ {}_I F_y \\ {}_I F_z \end{pmatrix} = -T \begin{pmatrix} \cos\psi \sin\theta \cos\varphi + \sin\psi \sin\varphi \\ \sin\psi \sin\theta \cos\varphi - \cos\psi \sin\varphi \\ \cos\theta \cos\varphi \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix}.$$

By solving this equation for φ, θ, and T, we can write the following transformation of the control inputs:

$$\theta^* = \arctan\left(\frac{\cos\tilde{\psi} \cdot {}_I F_x^* + \sin\tilde{\psi} \cdot {}_I F_y^*}{{}_I F_z^* - mg}\right), \tag{4}$$

$$\varphi^* = \arctan\left(\frac{\sin\tilde{\psi} \cdot {}_I F_x^* - \cos\tilde{\psi} \cdot {}_I F_y^*}{{}_I F_z^* - mg} \cos\theta^*\right), \tag{5}$$

$$T^* = \frac{mg - {}_I F_z^*}{\cos\theta^* \cos\varphi^*}. \tag{6}$$

Assuming that the attitude control's dynamics is fast and smooth enough, the second-order system block (attitude controller dynamics) and the control input transformations

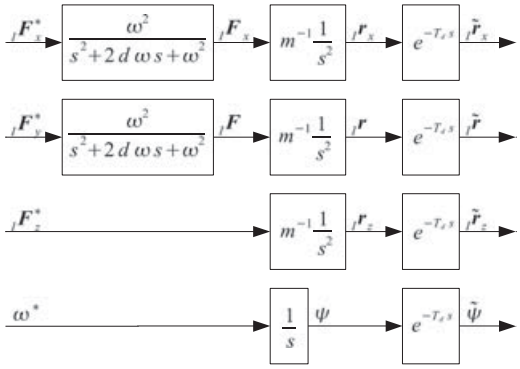


Figure 5. The four decoupled linear systems. They were obtained by transforming the system inputs of Figure 4 and exchanging the order of the nonlinear input transformation and the second-order system blocks. This can be justified by the smoothness of the nonlinear transformation and the high speed of the second-order system.

can be exchanged in order to obtain a new plant with the input ${}_I F_x^*$, ${}_I F_y^*$, ${}_I F_z^*$, and ω^*. This yields four decoupled linear systems that can be controlled separately (see Figure 5).

Because the subsystem for the yaw angle is simple and can be easily stabilized, a simple design will be sufficient: it outputs a constant angular velocity ω^* when the angular error exceeds a certain value. Thereby, large angular velocities that could destabilize the position of the helicopter can be avoided.

The position controllers are designed by means of the discrete LQG/LTR approach. The procedure is identical for all three position values, except that for the z-coordinate the second-order system is left out. Because of the limited computational power, the constant controller frequency is kept at roughly 20 Hz. This approximately matches the frequency of the SLAM pose estimates (around 15–30 Hz). For the Nyquist frequency, we take 7.5 Hz (half of the measurement frequency). It is possible that the measurement frequency temporarily falls beneath 15 Hz, but this generally does not last longer than some 100 ms.

The discrete system model is derived via the zero-order hold transformation of the continuous time model including the second-order system of the internal controller dynamics and the momentum law (double integration). After that, the time delay, approximated by a multiple of the sampling time, is added at the output of the model. Additionally, due to varying battery power and tilt angle calibrations, an output-error integrating part is added to the system. This eliminates steady-state errors induced by these varying factors.

Now the corresponding system matrices F, G, C, D required for the LQG/LTR can be computed. Applying the LQG/LTR procedure with feed-forward action yields the structure in Figure 6.

The Nyquist frequency (around 7.5 Hz) is strongly limiting the tuning process of the LQG/LTR. The resulting

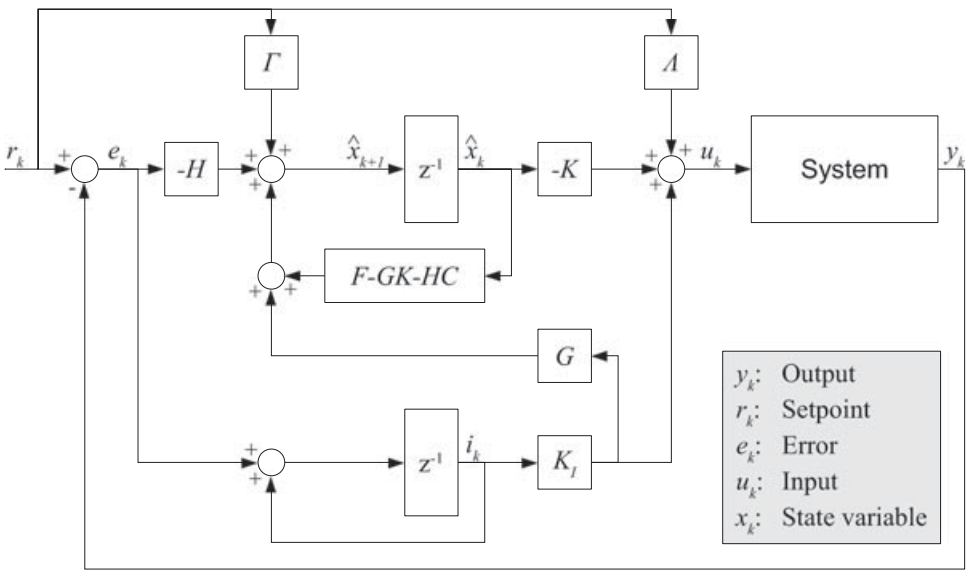


Figure 6. LQG/LTR controller structure. It results from the combination of a state observer and a state feedback controller. An integrating part and feed-forward action were included in order to obtain unity steady state gain and to render the observation error independent of the reference value. F, G, C are the discretized system matrices. The controller gains K, K_I, Γ, and Λ are obtained using the LQG/LTR procedure.

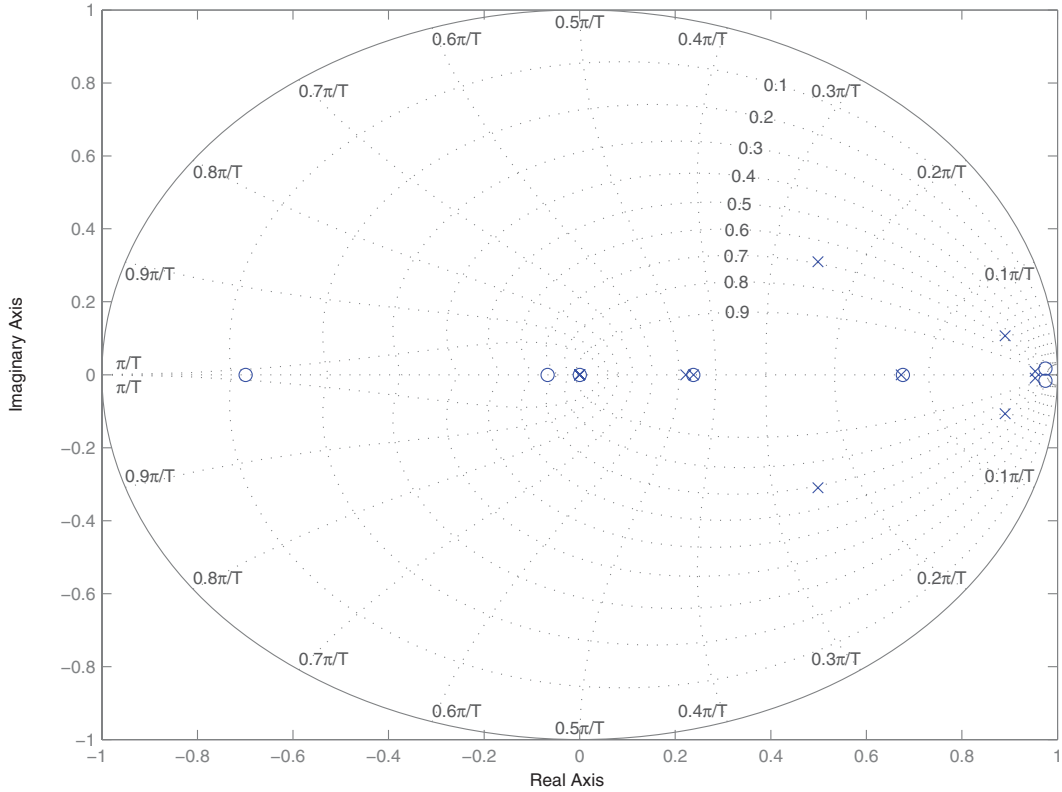


Figure 7. Pole-zero map of the closed-loop system. Except for the four poles induced by the time delay, all poles lie within 1–32 rad/s. No pole has a damping lower than 0.5. A zero situated at −7.2 is not displayed on the plot.

closed-loop system has its poles as in Figure 7. Except for the four poles induced by the time delay, all poles are between 1 and 32 rad/s. This enables the system to stabilize from an initial error of 1 m with a T_{90} time of around 1 s and 20% overshoot [Figure 8(a)]. At the expense of the performance, we attempt to maximize the robustness of the system in order to handle the modeling errors and external disturbance. However, to attain fast error correction at the beginning, a sufficiently large integrating part has to be maintained. Examining the Nyquist plot [Figure 8(b)], we can observe a phase margin of 27.7 deg and a gain margin of 5.5 dB, suggesting an acceptable robustness.

The main advantage of the LQG/LTR controller design is that it is a rather simple procedure to build linear stabilizing controllers with good robustness qualities and convenient tuning parameters. It also filters the measurements and estimates the velocity of the helicopter. Compared to the standard proportional–integral–derivative (PID) approach, it enables us to handle the nonnegligible time delay in order to obtain faster controllers.

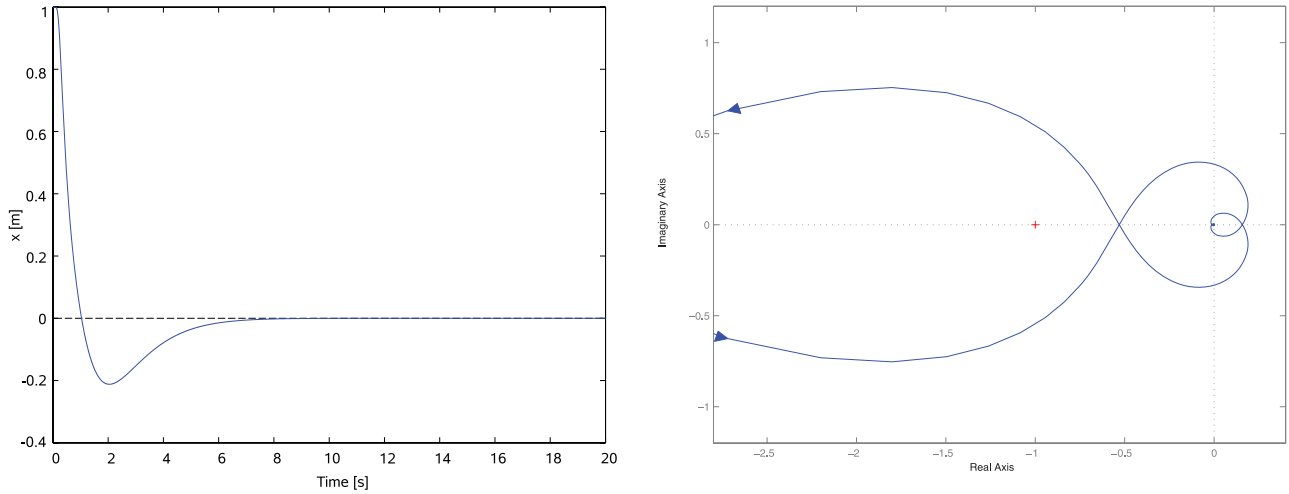
To ensure controllability of the x and y positions, we limit the force in the z direction to (m·g)/2. Otherwise, the total thrust T could go toward 0, disabling any control in the x and y directions.

7. FINAL SYSTEM STRUCTURE, FINAL IMPLEMENTATION

We use the Pelican quadrotor platform from Ascending Technologies (AscTec, N.D.). An onboard eight-bit micro controller combines all data in order to obtain an estimate for both tilt angles and the angular velocity around the vertical axis. Based on these estimates, a high-performance onboard controller is able to stabilize the tilt angles and the yaw rate at desired reference values. These can be sent by an external operator via an XBee digital radio.

Beneath the quadrotor, we mounted a 12-g USB-2 uEye UI-122xLE camera that gathers 752 × 480 pixel images with global shutter. We used a 190-deg-field-of-view lens to generate wide-angle images of the surroundings. Finally, both the visual SLAM algorithm and the controller run on a 1.6-GHz Intel Atom computer installed onboard the helicopter.

In the flow diagram (see Figure 9), the entire closed-loop system is represented. Based on different flight phases, the quadrotor is able to land or take off from the ground. Because the vision-based localization does not work when the helicopter is on the ground (the camera is too near to the floor), the takeoff is feasible only when the initial land patch



(a) (b)

Figure 8. (a) Time domain system response to an initial error of 1 m. The controller is able to correct the error with a T_{90} time of around 1 s and an overshoot of 20%. The performance is limited by the relatively slow measurement rate and the time delay of the system. (b) Nyquist plot of the system (open loop). Phase margin: 27.7 deg. Gain margin: 5.5 dB.

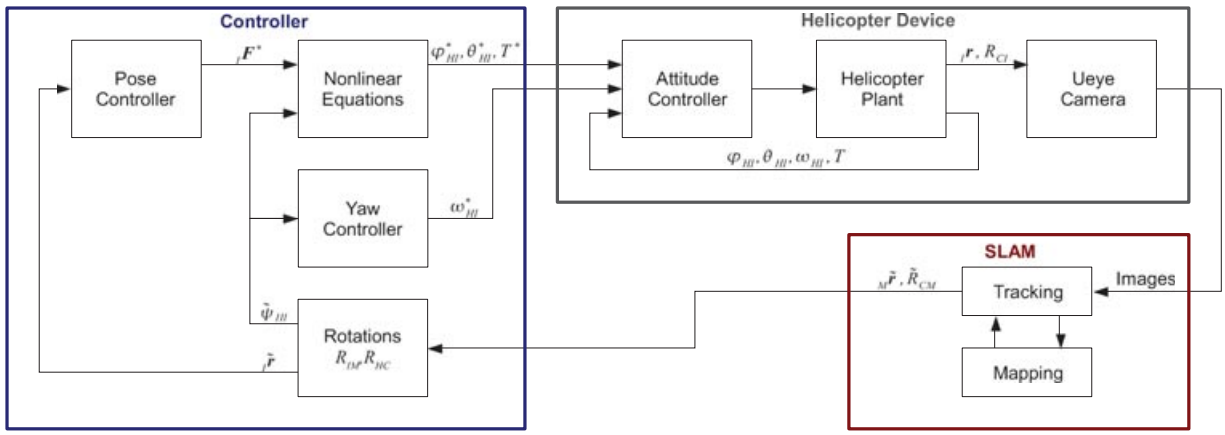


Figure 9. Closed-loop controller structure. The entire data flow is represented. Receiving the images from the mounted camera, the SLAM algorithm computes the estimate of the pose of the camera. This is transformed into the position and yaw angle of the helicopter in the inertial frame and fed to the controller. After the nonlinear control input transformations [Eqs. (4)–(6)], the reference values are sent back to the attitude controller on the quadrotor.

beneath it is already stored in the map. Giving increasing thrust, the helicopter can then fly blindly until it detects the map and stabilizes itself (the position can be tracked from a height of about 15 cm).

The problem of rotational drift is currently solved by realigning the inertial coordinate frame every 40 cm. For this, the helicopter has to be stabilized until its pose is approximately horizontal. This is done by observing the RMS value of the last 30 position errors (around 1.5 s). If this value is beneath a certain threshold (0.06 m), we can assume that the pose is horizontal (±0.02 rad in the tilt angles). This also leaves the mapping thread of the SLAM algorithm some time to expand the map. The issue of rotational drift is addressed in more detail in the next section.

8. EVALUATION OF THE VISUAL SLAM ACCURACY

For completeness, we analyze the visual SLAM algorithm's accuracy and behavior. We ran several tests in an external Vicon-motion-capture system to compare the SLAM framework to ground truth. This section also points out the fact that for controlling a MAV, it is sufficient to have an accurate local pose estimate and that there is no need for a globally consistent map. We remark that the tests in the

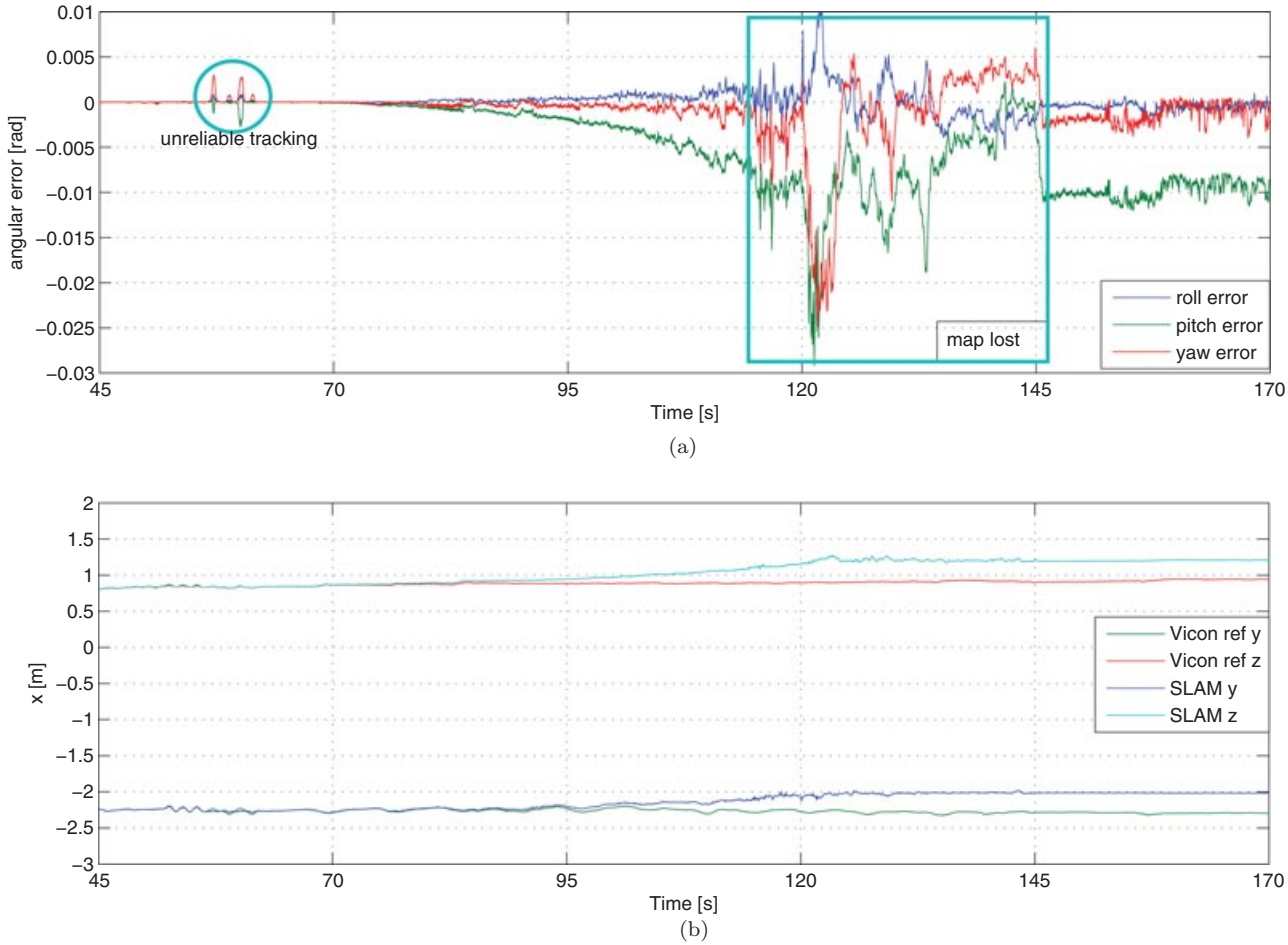


Figure 10. (a) Map drift represented in angular errors. Note that as we move in the positive x direction the angular error in pitch is most affected. From seconds 115 until 146, the map is lost and the data are corrupted. After, the map is recovered again. However, the angular error remains as before the map loss. Note that this plot also reveals unreliable tracking parts as shown around the 60th second. (b) Owing to the angular drift, the position experiences an error as well. For example, the drift toward negative roll leads to a higher z position compared to reality. The wrong angle estimation eventually leads to instability of the controller because of misalignment to the inertial system.

Vicon-motion-capture system were used only for ground truth comparisons and not to control the helicopter nor to gain quantitative knowledge for any control parameters.

Apart from the algorithm speed, there are three particular types of behaviors of the visual SLAM algorithm that are of interest when using it as input for a controller: the scale drift, the map drift, and localization failures.

The scale drift was handled by opportunely fusing the visual SLAM algorithm with the IMU data as described in our previous work (Nutzi et al., 2010). Whereas the scale drift affects the error measurement to the target, the map drift perturbs the pose itself. In the tests in the Vicon-motion-capture-system, we identified a nonnegligible rotational map drift so that the inertial coordinate frame is no longer correctly aligned. Rotations around the z axis can be adjusted by the yaw controller. Rotations around the x and y axes are more problematic. Because the error between the reference value and the actual position is no longer correctly decomposed into its x, y, and z values, this can lead to instability if not considered. In Figure 10(a) the rotational drift for the above-mentioned linear trajectory can be observed. Because the movement occurs in the x direction, the pitch-angle estimation is mostly affected by the drift. The same plot also reveals when the tracking thread lost the map and when unreliable tracking occurred. In Figure 10(b), the resulting deviations in position are depicted. The drift in negative pitch, for example, leads to a higher estimated z value than it is in reality.

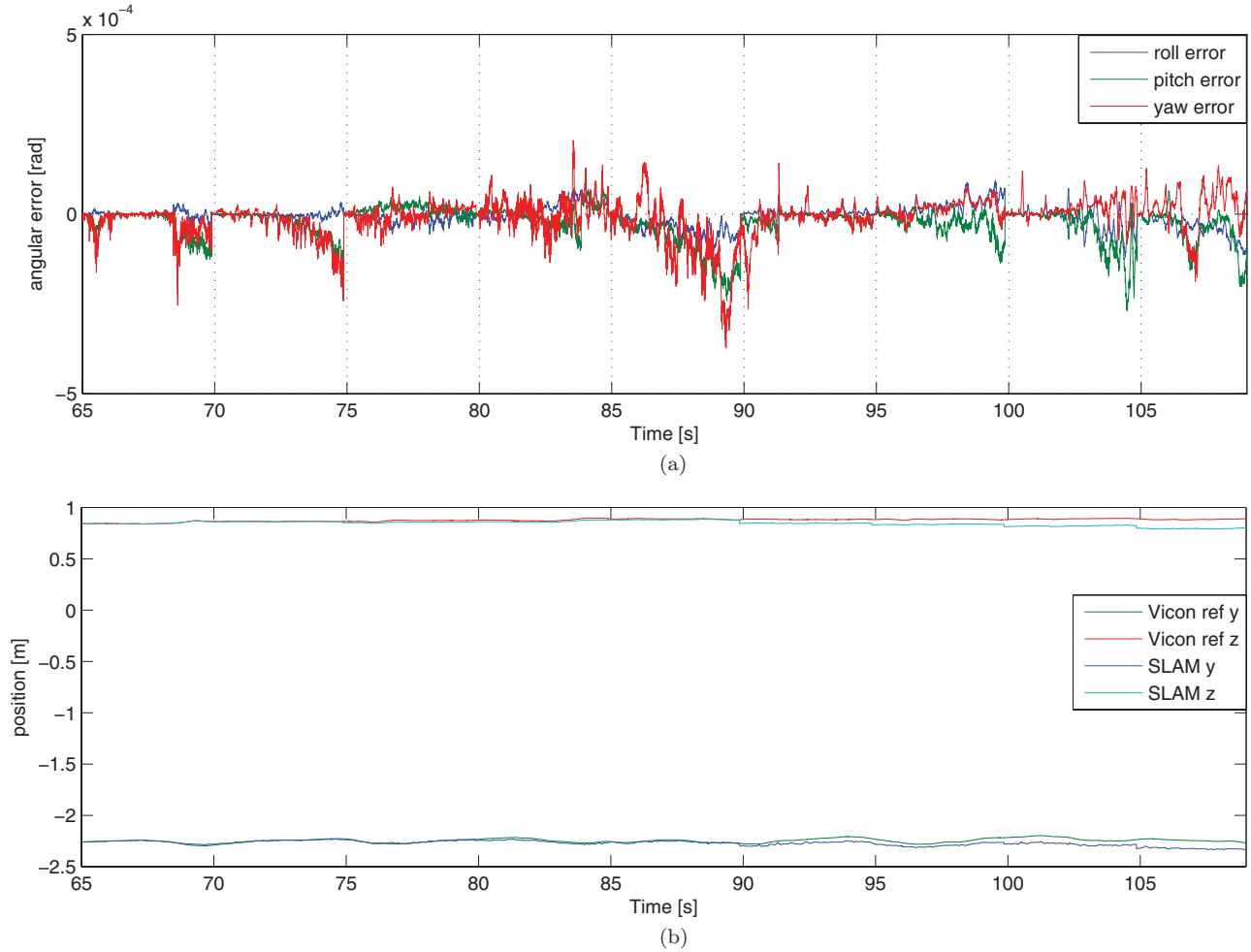


Figure 11. (a) To overcome the map drift problem, correction steps are applied each $\Delta t = 5$ s. The helicopter is set to hover mode while inertial data are gathered. Based on the sensor information, the map is realigned to the inertial frame. On our platform we applied the correction step at a fixed traveled distance Δd rather than making it dependent on time. (b) By eliminating the angular drift in the map, the position drift is minimized as well.

To overcome the problem of map drift, we implemented a very simple correction step in our controller. That is, we read the inertial measurements in hovering mode and realign the SLAM map according to the inertial frame. We decided to apply this step every time a defined distance Δd has been traveled. Making this step independent of time, we save computation power while hovering. In Figure 11(a), the effect of such correction steps is shown. Note that to facilitate the understanding of this figure, we applied the correction at a fixed time difference $\Delta t = 5$ s rather than at a fixed distance, as we did on the real system. The angular error is reset every Δd, and the map drift in position is minimized as shown in Figure 11(b). These correction steps do not improve the consistency of the global map. However, they assure accurate local pose estimations. This is sufficient for robust MAV control.

Short failure periods—as seen in Figure 10(a) around the 60th second—are unavoidable in real systems because they originate from various sources (extensive occlusion, temporarily wrong data association, etc.). However, being short enough, they do not destabilize the controller. Position spikes are filtered in the observer model of the controller. Periods of bad tracking are recognized by the visual SLAM algorithm. To overcome short periods of bad tracking, we implemented a safety procedure providing an average thrust computed as the mean thrust every time the helicopter was in hovering mode:

$$\mathbf{T} = \sum_{t=1}^{N} \frac{\mathbf{T}_t}{N}, \tag{7}$$

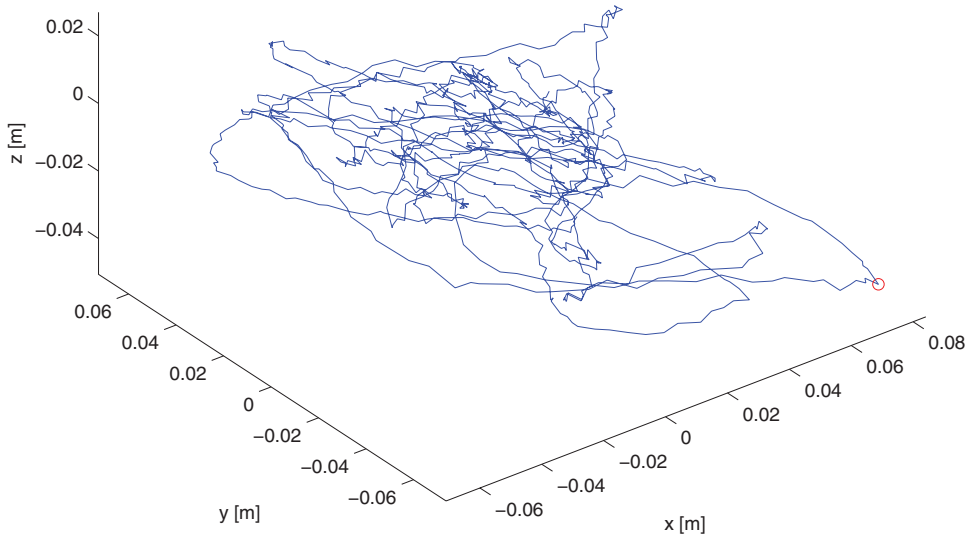


Figure 12. Position error while hovering during 60 s. The RMS value of the position error is 2.89 cm in x, 3.02 cm in y, and 1.86 cm in z. The maximum runaway has an absolute error value of 11.15 cm (marked with a red o).

where $\mathbf{T}_t$ is the thrust applied at time t when the inertial measurements indicated hovering mode. The more hovering periods the helicopter had when the tracking failure occurs, the more stable it is during tracking failures and the longer failure periods it can compensate.

9. RESULTS AND DISCUSSION

With the techniques mentioned in the preceding section, we were able to navigate robustly our MAV using our vision-based controller. To test the platform, a setup was constructed in which the helicopter was secured by two nylon wires in a large indoor laboratory environment. These wires are visible in some of the attached videos. For outdoor experiments we sometimes used a fishing rod for security reasons. However, when the helicopter was flying at high altitudes we did not use any security wires and just relied on the remote controller as a backup in case of failures.

For the experimental tests, a downward-looking camera is mounted on a quadrocopter from Ascending Technologies. As the camera, we use a Point-Grey USB Firefly camera with a WVGA resolution of 752 × 480 pixels and a global shutter. The camera faces the ground with a 150-deg-field-of-view lens. The images are fed to the visual SLAM algorithm described in Section 4.

When hovering, the quadrotor is very stable. In Figure 12 the flight path for a 60-s hovering can be seen. Because the mapping thread goes almost into sleep mode once a region has been explored, the computational power can be fully used for tracking and controlling the quadrotor. In addition, the map will never get corrupted in this state and the localization quality is guaranteed. Overall, the position error during 60 s in hovering has an RMS value of 2.89 cm in x, 3.02 cm in y, and 1.86 cm in z, which yields an absolute RMS error value of 4.61 cm (see Figure 13).

We tested the performance of the controller and the accuracy of the underlying model by applying a step input to the real system. No quantitative tests using constant or impulse disturbances were applied. This is because of the lack of appropriate equipment (wind tunnel and corresponding measurement devices). However, we refer to the qualitative tests shown in the corresponding videos. We applied a step input by setting a desired position 35 cm away from the current hovering position. This is actually the action the MAV will perform when following waypoints.

The controller is able to correct the error with a T_{90} time of around 1 s as modeled in Section 6 and an overshoot of 50% as compared to the modeled 20%. Figure 14 depicts the evolution of the step input applied to the MAV.

The overshoot is larger than predicted by the model in Figure 8(a); however, it is sufficient for robust MAV

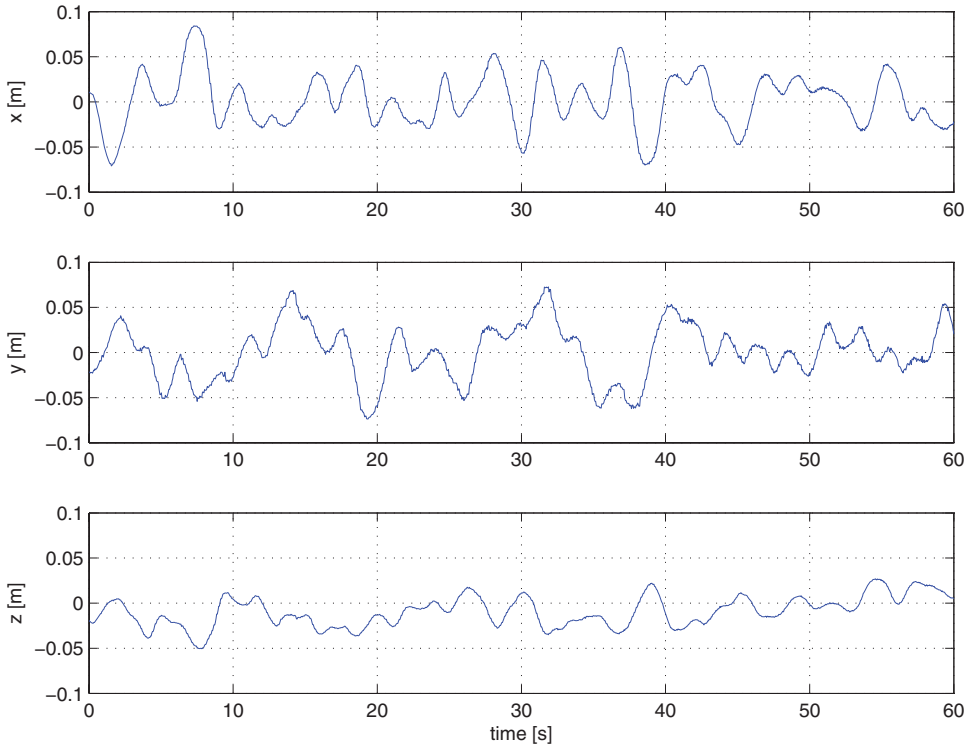


Figure 13. Position error in the x, y, and z positions. The value remains ± 10 cm. The z position is more accurate than the x and y positions.

navigation. We note here that the large overshoot is caused by the integral block in the controller structure. Also because of this, windup issues have to be considered as soon as the MAV is prevented from reaching a goal position (obstacle, MAV fixed to a cord, etc.).

To test the waypoint following ability, we applied a set of step inputs and verified the controller performance. Figure 15 shows that the MAV can robustly follow one waypoint after the other.

The platform is now able to fly to desired setpoints. For this, the path is split into waypoints. The distance between them is chosen so that the helicopter can realign the orientation of the inertial coordinate frame at each waypoint. Here, a little higher RMS value is obtained as in the hovering mode (see Figure 16). The map consisted of seven key frames at the start. While expanding the map, the SLAM algorithm has processed more than 40 additional key frames. At each waypoint, the helicopter stabilizes itself and waits until the RMS value is small enough (10 cm in absolute error value) to realign the map. At the same time, this leaves some time to the SLAM algorithm to process new key frames and expand the map. This shows the idea of having only a locally consistent map to control the helicopter. The realignment leads to a correct map representation in the close neighborhood. However, it distorts the map positions currently unobservable by the camera. The map might not appear to be globally consistent, but nevertheless the vehicle is well controlled at any position.

At some points, when the helicopter flies from waypoint to waypoint, a notable overshoot can be observed. This is due to the integrating action of the controller, which is necessary to correct steady state errors. It does introduce additional overshoot and reduces the robustness of the system. However, a satisfying trade-off was found.

In Figure 17 the map that was built during the flight can be seen. It is composed of 52 key frames and 4,635 map points. It represents an approximate surface of 15 m^2. The quadrotor will always localize itself correctly with respect to the neighboring local maps, so that if the map drifts, this will not influence the control quality.

Some failure modes also have to be mentioned. Owing to the lack of features or to varying illumination, it is possible that the tracking thread cannot find enough features. This would disable any localization of the vehicle. Also, if the map becomes too large, the mapping thread may use too much computational power for the global bundle adjustment, leading to a greater time delay and slower rate of the pose estimates until the controlled system becomes unstable.

The achieved results show that our platform can autonomously fly through a large unknown indoor environment with a high degree of accuracy. The system is

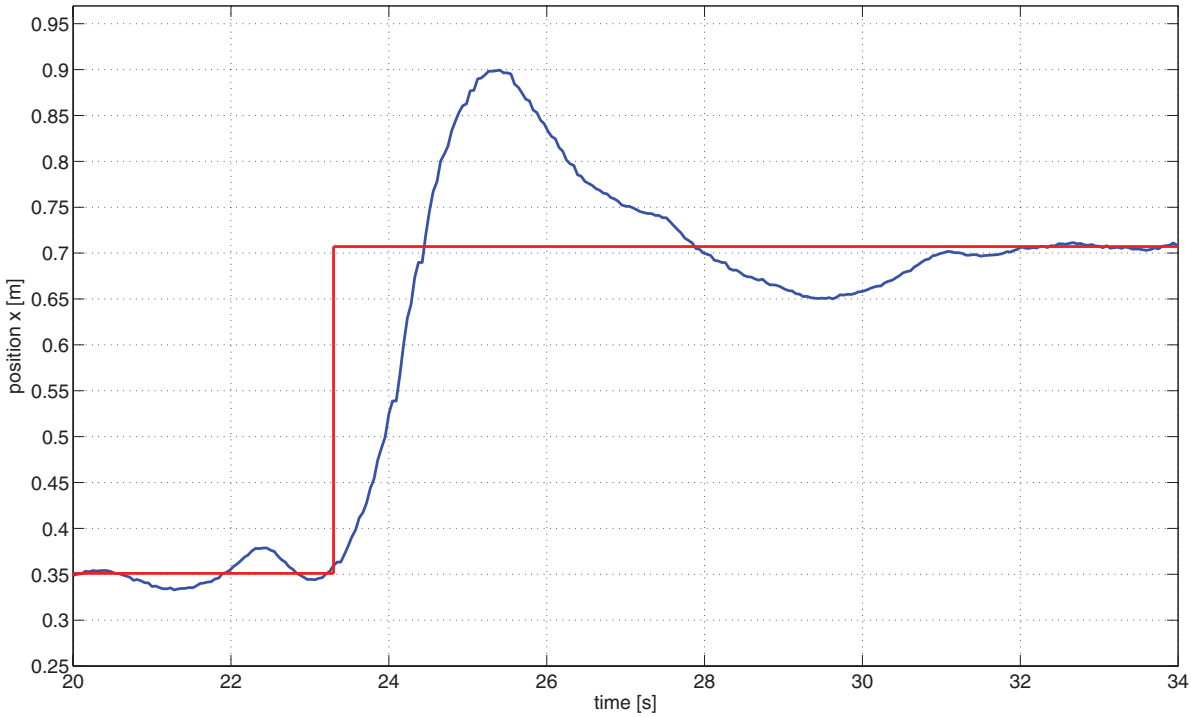


Figure 14. Step input applied to the real system. The controller is able to correct the error with a T_{90} time of around 1 s as modeled in Section 6 and an overshoot of 50% as compared to the modeled 20%. The overshoot is caused by the integrator block in the controller structure.

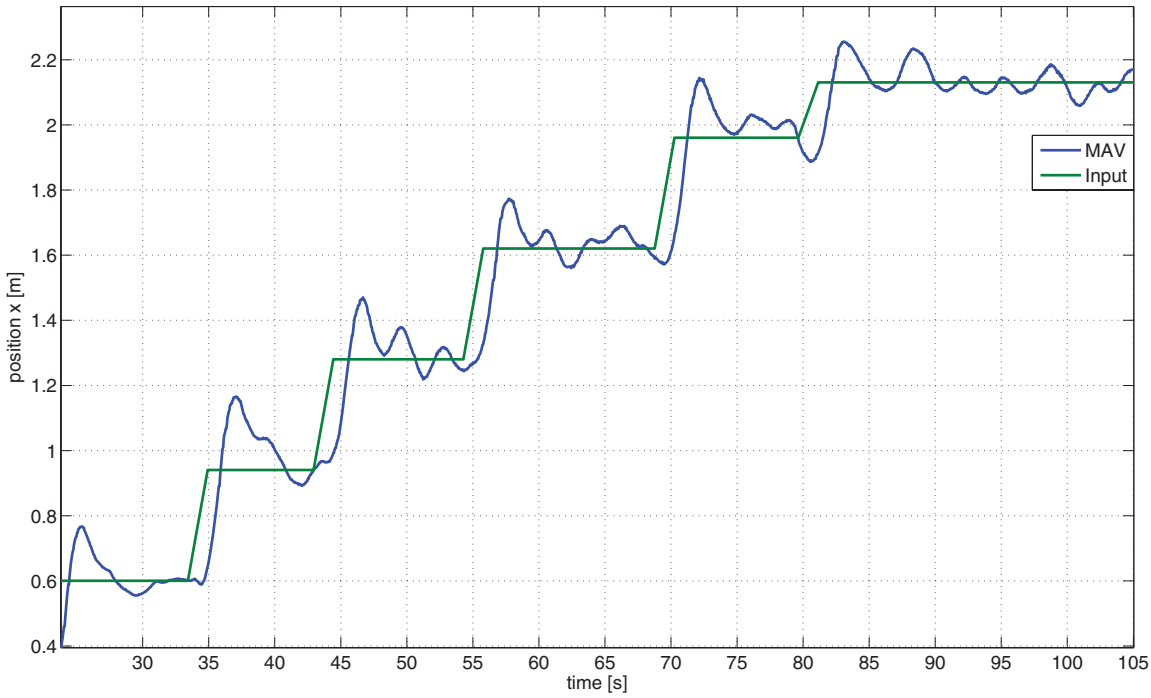


Figure 15. Several step inputs are applied one after the other to the real system. This is an intermediate step toward an ongoing waypoint following. The plot shows that the system is able to follow robustly the applied sequence and is thus capable of following any trajectory, as we show later in this paper.

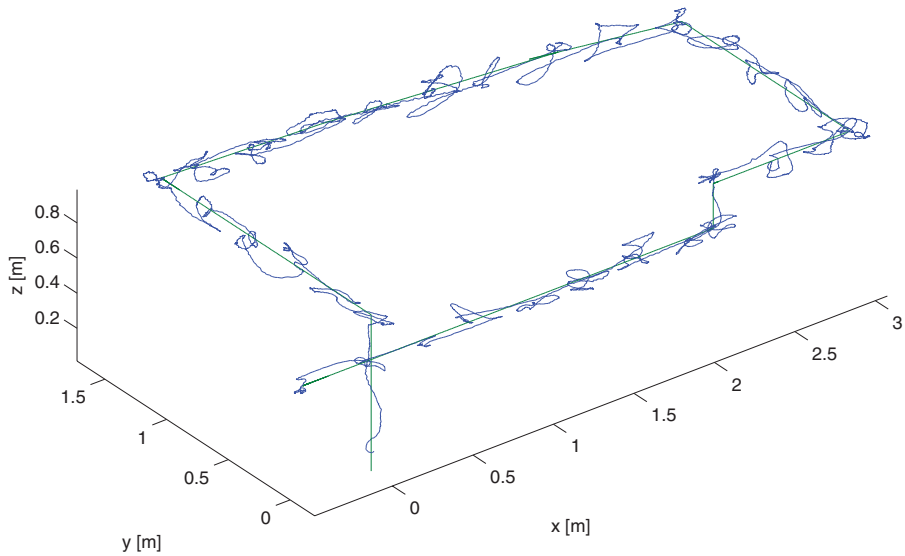


Figure 16. Path that the helicopter has flown. This does not represent ground truth. It is the pose estimation of the SLAM algorithm. However, the attitude of the helicopter can be observed while successfully flying a rectangular loop and landing on the ground. The RMS value of the position error is 9.95 cm in x, 7.48 cm in y, and 4.23 cm in z. The path has a total length of a little bit more than 10 m in an area of $3.5 \times 2 \times 1$ m^3.

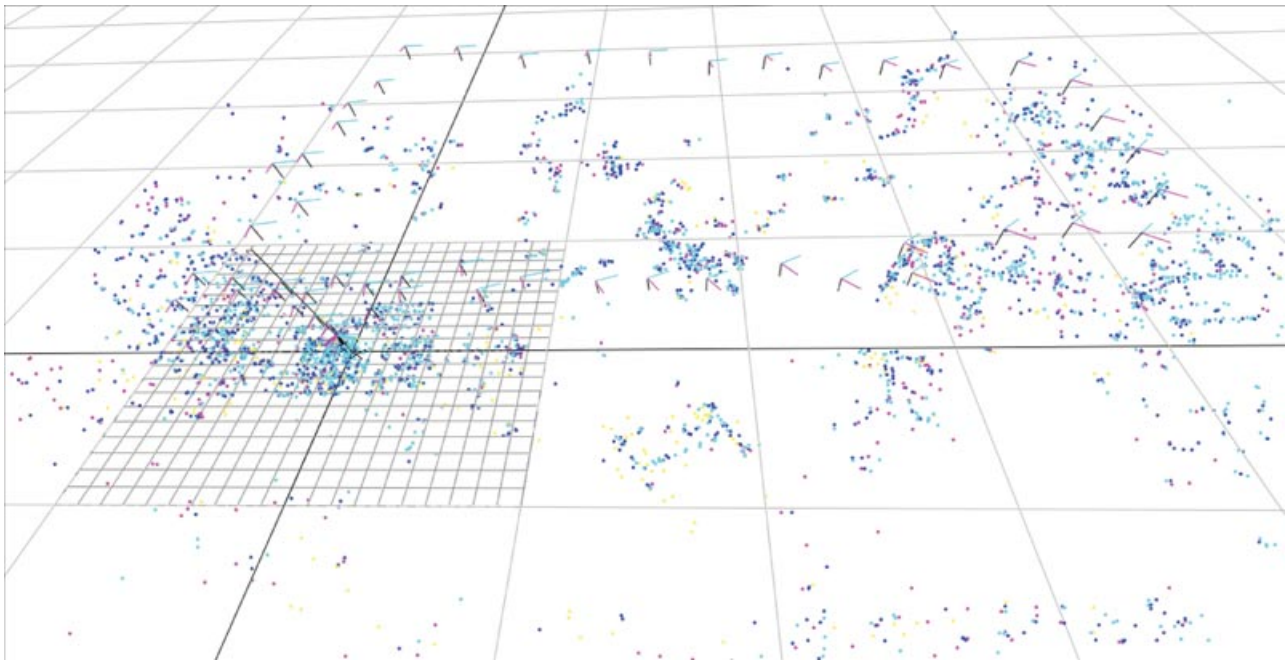


Figure 17. 3D view of the built map. It contains 4,635 map points, observed in 52 different key frames.

robust against external disturbances and can handle modeling errors. Outdoor tests confirm the controller's robustness, which was able to handle considerable side winds and turbulence.

10. CONCLUSION

This paper presented a vision-based MAV control approach. The pose was estimated by means of a monocular SLAM algorithm with a precision of a few centimeters.

This was then used to stabilize the position of the vehicle. Based on a control input transformation and on the linear LQG/LTR procedure, a controller was designed. The resulting platform successfully managed to hover and follow desired setpoints in an indoor laboratory. For this, it does not need any prior information on the environment. After the initialization, a map of the surroundings was built incrementally, wherein the MAV was able to localize itself without any time drift. The vehicle can control its position up to a few centimeters of error (RMS around 2–4 cm). We showed that this is possible by having only a local consistent map, making the controller largely independent of map and scale drift. All calculations including SLAM and controller are running in real time and online while the MAV is flying. No offline or preprocessing is done. Using the proposed strategy, our helicopter is able to perform all basic flight maneuvers such as autonomous take off, set-point following, and landing.

We note that the described approach needs short-period hovering slots to adjust the local map with the gravity vector. This way the controller is largely independent of map and scale drift. However, continuous flight without pauses is still an open issue. Continuous flight in the sense of continuous operation in unknown environments is achieved with this approach.

For future systems we are strongly motivated toward modular sensor fusion. We aim at a modular approach in order to be independent of the underlying vision algorithm. This way, we can select the (computationally) most efficient solution for good onboard performance. Also modular, or loosely coupled, solutions are computationally less expensive and thus more suited for MAVs. Additionally, vision SLAM algorithms depend on a (local) map. If this map is lost for any reason, the algorithm enters an expensive recovery mode. This issue should be tackled in order to have a good state estimation of the MAV at any given time.

11. APPENDIX: INDEX TO MULTIMEDIA EXTENSIONS

The videos are available as Supporting Information in the online version of this article.

Extension	Media type	Description
1	Video	Autonomous takeoff, hovering, set-point following, and landing using the approach described in this paper. Disturbances are applied.
2	Video	Autonomous takeoff using one landmark, hovering, set-point following, and landing using the approach described in this paper. The MAV navigates into and lands in previously unknown area.
3	Video	EMAV competition in Delft, Holland, 2009. The MAV navigates through a window using the approach described in this paper
4	Video	Helicopter hovering outdoors above the asphalt (low contrast, repetitive structure) and also in presence of wind.
5	Video	Autonomous outdoor hovering in windy conditions above a repetitive texture ground. Shows failure mode because of mismatches on the repetitive texture.
6	Video	Helicopter flying above an (abandoned) village. The environment is reconstructed and textured with the aid of a meshing approach.
7	Video	Autonomous vision–based outdoor hovering compared to GPS-based hovering.

Video extension 1 shows autonomous takeoff, hovering, set-point following, and landing using the approach described in this paper. Furthermore, you can also see that the helicopter is heavily perturbed by a human to show stability against turbulences. Notice that the two cables are used only for security reasons in case something goes wrong. Thus, they do not constrain the motion of the helicopter. In the last third of the video, we show successful navigation through a circular window of 1-m diameter. For safety, security, and rescue robotics (SSRR) this is a vital ability in order to penetrate unknown areas difficult to access for humans.

Video extension 2 shows another demonstration of takeoff, hovering, set-point following, and landing using our approach. In this video, you can also see both the helicopter and the camera's view with overlaid features during the all operation. The difference with the previous video is that here we also experimented with a blob-based takeoff and landing. The blob is the black-and-white circle at the beginning and at the end of the experiment. This was done for our participation in the European MAV competition (see next video), in which the rules required the ability to do high-precision takeoff and landing from a given pattern. This is, however, not part of this paper. In the current paper, we do not use any external or artificial landmark but just natural features. The combination of the high-precision takeoff and landing together with the capability of exploring unknown terrain using natural features shows the potential for SSRR missions in an SSRR scenario, much like the landing blob in the video, a defined object can be searched for during the SLAM-based navigation and then used as

precision landing target. The interested reader can find a detailed description of the blob-based pose estimation in our previous work (Eberli, Scaramuzza, Weiss, & Siegwart, 2010).

Video extension 3 shows our helicopter during the EMAV'09 competition that took place in Delft, Holland, in September 2009 and where our team ranked second. Our helicopter was the only autonomous vehicle using vision. The task consisted of taking off, approaching a small apartment, and passing through its window. This video demonstrates autonomous takeoff, hovering, and set-point following, using the approach described in this paper. Notice that toward the end of the video the helicopter was able to fly through the window. This was possible because in this case we visually mapped the environment just before the competition. Again, notice that the fishing rod is used only for security reasons because we were operating in an open, public space. Also observe that—because the competition took place in a gymnasium—there was no texture on the floor. Textureless environments are a known limitation of any vision-based algorithm. To overcome this problem, we added some sparse features. The features consisted of pieces of adhesive tape stuck on some bands. Please notice that we could have used any other natural features (such as stones thrown on the floor) but we were not allowed to dirty the floor due to the strict rules of the competition. In our outdoor experiments, stones indeed proved to be robust features. This gives motivation to use our approach in disaster areas such as partially destroyed houses after earthquakes or in similar SSRR missions outdoors.

Video extension 4 shows our helicopter hovering outdoors above the asphalt and also in the presence of wind. Compared to stones (i.e., ballast), asphalt is a very challenging terrain due to the very low contrast and the lack of very distinctive features. Notice, indeed, that the vehicle starts to oscillate when it gets too high. This is because—with the height—the resolution of the features on the ground and, thus, the number of features decrease. In essence, the more cluttered the environment is, the more contrast we have and thus the more robust the navigation is. Again, this motivates SSRR missions in cluttered disaster areas. Notice also that wind was present and that the fishing rod was again used for security.

Video extension 5 shows outdoor hovering in windy conditions above a repetitive-texture ground. As you can see, the ground consists of small self-similar checkers. At some point the helicopter becomes unstable because of mismatches among the corners of the checkers, which all look the same. Repetitive texture is a known limitation of vision-based algorithms.

Video extension 6 shows an outdoor mapping result. Here the helicopter was flying above a village (current military area) near Zurich. The average height of the helicopter during the flight was 15 m. The flight represents a typical reconnaissance mission task monitoring an urban environment. This video shows the feature tracking and mapping process. Texture was added to the 3D map. This was done by first performing a Delaunay triangulation and then projecting the texture onto the 3D mesh. This is, however, not part of this paper. The interested reader can find a detailed description of the textured-3D mapping in our previous work (Weiss, Achtelik, Kneip, Scaramuzza, & Siegwart, 2010).

Video extension 7 shows another outdoor experiment. In this video, you can see two helicopters in hovering mode. One uses GPS for stabilization, and the other one only vision, using the method described in this paper.

ACKNOWLEDGMENTS

The authors thank in particular Mike Bosse, from CSIRO, for his fruitful input about data analysis and Sergei Lupashin from ETH Zurich, who made it possible for us to use an external Vicon-motion-capture system for ground truth comparisons. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 231855 (sFly).

REFERENCES

Abdelkrim, N., Aouf, N., Tsourdos, A., & White, B. (2008, June). Robust nonlinear filtering for INS/GPS UAV localization. In Mediterranean Conference on Control and Automation, Ajaccio, France (pp. 695–702).

Achtelik, M., Bachrach, A., He, R., Prentice, S., & Roy, N. (2009, April). Stereo vision and laser odometry for autonomous helicopters in GPS denied indoor environments. In Proceedings of the SPIE Unmanned Systems Technology XI, Orlando, FL.

Ahrens, S., Levine, D., Andrews, G., & How, J. (2009, May). Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments. In International Conference on Robotics and Automation, Kobe, Japan.

Altug, E., Ostrowski, J., & Mahony, R. (2002, May). Control of a quadrotor helicopter using visual feedback. In International Conference on Robotics and Automation, Washington, DC (pp. 72–77).

Artieda, J., Sebastian, J. M., Campoy, P., Correa, J. F., Mondragón, I. F., Martínez, C., & Olivares, M. (2008). Visual 3-D SLAM from UAVs. Journal of Intelligent and Robotic Systems, 55(4–5), 299–327.

AscTec. (N.D.) Ascending Technologies GmbH. http://www.asctec.de. Accessed August 21, 2011.

Bachrach, A., He, R., & Roy, N. (2009a). Autonomous flight in unknown indoor environments. International Journal of Micro Air Vehicles, 1(4), 277–298.

Bachrach, A., He, R., & Roy, N. (2009b, September). Autonomous flight in unstructured and unknown indoor environments. in European Conference on Micro Aerial Vechicles (EMAV), Delft, The Netherlands.

Belloni, G., Feroli, M., Ficola, A., Pagnottelli, S., & Valigi, P. (2007, September). An autonomous aerial vehicle for unmanned security and surveillance operations: Design and test. In IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR), Rome, Italy.

Birk, A., Wiggerich, B., Bulow, H., Pfingsthorn, M., & Schwertfeger, S. (2011). Safety, security, and rescue missions with an unmanned aerial vehicle (UAV): Aerial mosaicking and autonomous flight at the 2009 European Land Robots trials (ELROB) and the 2010 Response Robot Evaluation Exercises (RREE). Journal of Intelligent and Robotic Systems, 64, 57–76.

Bouabdallah, S., & Siegwart, R. (2005, April). Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In International Conference on Robotics and Automation, Barcelona, Spain (pp. 2247–2252).

Cai, G., Cai, A., Chen, B., & Lee, T. (2008, September). Construction, modeling and control of a mini autonomous UAV helicopter. In International Conference on Automation and Logistics, Qingdao, China (pp. 449–454).

Castillo, P., Lozano, R., & Dzul, A. (2004, October). Stabilization of a mini-rotorcraft having four rotors. In International Conference on Intelligent Robots and Systems, Sendai, Japan (pp. 2693–2698).

Chen, J., & Dawson, D. (2006, December). UAV tracking with a monocular camera. In Conference on Decision and Control, San Diego, CA.

Cheviron, T., Hamel, T., Mahony, R., & Baldwin, G. (2007, April). Robust nonlinear fusion of inertial and visual data for position, velocity and attitude estimation of UAV. In International Conference on Robotics and Automation, Rome, Italy (pp. 2010–2016).

Davison, A., Reid, I., Molton, N., & Strasse, O. (2007). MonoSLAM: Real-time single camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(6), 1052–1067.

Dhaliwal, S., & Ramirez-Serrano, A. (2009, November). Control of an unconventional VTOL UAV for search and rescue operations within confined spaces based on the Marc control architecture. In IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR), Denver, CO.

Eberli, D., Scaramuzza, D., Weiss, S., & Siegwart, R. (2010). Vision based position control for MAVs using one single circular landmark. Journal of Intelligent and Robotic Systems, 61(1–4), 495–512.

Fowers, S., Lee, D., Tippets, B., Lillywhite, K., Dennis, A., & Archibald, J. (2007, June). Vision aided stabilization and the development of a quad-rotor micro UAV. In International Symposium on Computational Intelligence in Robotics and Automation, Jacksonville, FL (pp. 143–148).

Garratt, M., & Chahl, J. (2008). Vision-based terrain following for an unmanned rotorcraft. Journal of Field Robotics, 25(4–5), 284–301.

Goodrich, M., Cooper, J., Adams, J., Humphrey, C., Zeeman, R., & Buss, B. (2007, September). Using a mini-UAV to support wilderness search and rescue: Practices for human–robot teaming. In IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR), Rome, Italy.

Hamel, T., Mahony, R., & Chriette, A. (2002, May). Visual servo trajectory tracking for a four rotor VTOL aerial vehicle. In International Conference on Robotics and Automation, Washington, DC (pp. 2781–2786).

Herisse, B., Russotto, F., Hamel, T., & Mahony, R. (2008, September). Hovering flight and vertical landing control of a VTOL unmanned aerial vehicle using optical flow. In International Conference on Intelligent Robots and Systems, Nice, France (pp. 801–806).

How, J., Bethke, B., Frank, A., Dale, D., & Vian, J. (2008). Real-time indoor autonomous vehicle test environment. IEEE Control Systems Magazine, 28(2), 57–64.

Hrabar, S., Sukhatme, G., Corke, P., Usher, K., & Roberts, J. (2005, August). Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In International Conference on Intelligent Robots and Systems, Edmonton, AB, Canada (pp. 3309–3316).

Klein, G., & Murray, D. (2007, November). Parallel tracking and mapping for small AR workspaces. In International Symposium on Mixed and Augmented Reality, Nara, Japan (pp. 225–234).

Klose, S., Wang., J., Achtelik, M., Panin, G., Holzapfel, F., & Knoll, A. (2010, October). Markerless, vision-assisted flight control of a quadrocopter. In IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan.

Ludington, B., Johnson, E., & Vachtsevanos, G. (2006). Augmenting UAV autonomy: Vision-based navigation and target tracking for unmanned aerial vehicles. IEEE Robotics and Automation Magazine, 13(3), 63–71.

Lupashin, S., Schöllig, A., Sherback, M., & D'Andrea, R. (2010, May). A simple learning strategy for high-speed quadrocopter multi-flips. In International Conference on Robotics and Automation, Anchorage, AK.

Maza, I., Caballero, F., Capitán, J., de Dios, J. M., & Ollero, A. (2011). Experimental results in multi-UAV coordination for disaster management and civil security applications. Journal of Intelligent and Robotic Systems, 62(1–4), 563–585.

Mejias, L., Usher, K., Roberts, J., & Corke, P. (2006, October). Two seconds to touchdown—Vision-based controlled forced landing. In International Conference on Intelligent Robots and Systems, Beijing, China (pp. 3527–3532).

Mellinger, D., Michael, N., & Kumar, V. (2010, December). Trajectory generation and control for precise aggressive maneuvers with quadrotors. In International Symposium on Experimental Robotics (ISER), Delhi and Agra, India.

Mellinger, D., Shomin, M., Michael, N., & Kumar, V. (2010, November). Cooperative grasping and transport using multiple quadrotors. In International Symposium on Distributed Autonomous Robotic Systems (DARS), Lausanne, Switzerland.

Merino, L., Caballero, F., de Dios, J. M., Ferruz, J., & Ollero, A. (2006). A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires. Journal of Field Robotics, 23(3), 165–184.

Michael, N., Fink, J., & Kumar, V. (2010). Cooperative manipulation and transportation with aerial robots. Autonomous Robots, 30(1), 73–86.

Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The GRASP multiple micro UAV testbed. IEEE Robotics and Automation Magazine, 17(3), 56–65.

Nutzi, G., Weiss, S., Scaramuzza, D., & Siegwart, R. (2010). Fusion of IMU and vision for absolute scale estimation in monocular SLAM. Journal of Intelligent and Robotic Systems, 61(1–4), 287–299.

Park, S., Won, D., Kang, M., Kim, T., Lee, H., & Kwon, S. (2005, August). RIC (robust internal-loop compensator) based flight control of a quad-rotor type UAV. In International Conference on Intelligent Robots and Systems, Edmonton, AB, Canada (pp. 3542–3547).

Peng, K., Dong, M., Chen, B., Cai, G., Lum, Y., & Lee, T. (2007, July). Design and implementation of a fully autonomous flight control system for a UAV helicopter. In Chinese Control Conference, Zhangjiajie, China (pp. 662–667).

Proctor, A., & Johnson, E. (2004, August). Vision-only aircraft flight control methods and test results. In AIAA Guidance, Navigation, and Control Conference, Providence, RI.

Rosten, E., & Drummond, T. (2005, October). Fusing points and lines for high performance tracking. In IEEE International Conference on Computer Vision, Beijing, China.

Ruffier, F., & Franceschini, N. (2004, April). Visually guided micro-aerial vehicle: Automatic take off, terrain following, landing and wind reaction. In International Conference on Robotics and Automation, New Orleans, LA (pp. 2339–2346).

Saripalli, S., Montgomery, J., & Sukhatme, G. (2002, May). Vision-based autonomous landing of an unmanned aerial vehicle. In International Conference on Robotics and Automation, Washington, DC (pp. 2799–2804).

Strasdat, H., Montiel, J., & Davison, A. J. (2010, May). Real-time monocular SLAM: Why filter? In International Conference on Robotics and Automation, Anchorage, AK.

Templeton, T., Shim, D., Geyer, C., & Sastry, S. (2007, April). Autonomous vision-based landing and terrain mapping using an MPC-controlled unmanned rotorcraft. In International Conference on Robotics and Automation, Rome, Italy (pp. 1349–1356).

Valenti, M., Bethke, B., Fiore, G., & How, J. P. (2006, August). Indoor multivehicle flight testbed for fault detection, isolation and recovery. In AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO.

Weiss, S., Achtelik, M., Kneip, L., Scaramuzza, D., & Siegwart, R. (2010). Intuitive 3D maps for MAV terrain exploration and obstacle avoidance. Journal of Intelligent and Robotic Systems, 61(1–4), 473–493.

Wenzel, K. E., Masselli, A., & Zell, A. (2010, June). Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle. In UAV'10 3rd International Symposium on Unmanned Aerial Vehicles, Dubai, United Arab Emirates.

Yun, B., Peng, K., & Chen, B. (2007, May). Enhancement of GPS signals for automatic control of a UAV helicopter system. In International Conference on Robotics and Automation, Guangzhou, China (pp. 1185–1189).

Zingg, S., Scaramuzza, D., Weiss, S., & Siegwart, R. (2010, May). MAV navigation through indoor corridors using optical flow. In IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK (pp. 3361–3368).

Zufferey, J., & Floreano, D. (2006). Fly-inspired visual steering of an ultralight indoor aircraft. IEEE Transactions on Robotics, 22(1), 137–146.