2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Vision Based MAV Navigation in Unknown and Unstructured Environments

Michael Blösch, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart
Autonomous Systems Lab
ETH Zurich

Abstract— **Within the research on Micro Aerial Vehicles (MAVs), the field on flight control and autonomous mission execution is one of the most active. A crucial point is the localization of the vehicle, which is especially difficult in unknown, GPS-denied environments. This paper presents a novel vision based approach, where the vehicle is localized using a downward looking monocular camera. A state-of-the-art visual SLAM algorithm tracks the pose of the camera, while, simultaneously, building an incremental map of the surrounding region. Based on this pose estimation a LQG/LTR based controller stabilizes the vehicle at a desired setpoint, making simple maneuvers possible like take-off, hovering, setpoint following or landing. Experimental data show that this approach efficiently controls a helicopter while navigating through an unknown and unstructured environment. To the best of our knowledge, this is the first work describing a micro aerial vehicle able to navigate through an unexplored environment (independently of any external aid like GPS or artificial beacons), which uses a single camera as only exteroceptive sensor.**

I. Introduction

In the past years, micro aerial vehicles (MAVs) strongly gained in autonomy. This was motivated through the very wide field of applications for these little platforms. Commonly associated keywords are: search and rescue, exploration, surveillance, agriculture and inspection.

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 (3D orientation of the helicopter) of the MAV and one superposed controller for its 3D position. Most attitude controller use the feedback from an onboard inertial measurement unit (IMU). With this good performance can often be obtained with a simple PD-controller design, but also more sophisticated design techniques have been applied [1], [2], [3]. E.g. Bouabdallah et al. [4] analyzed the application of two different control techniques "Sliding-Mode" and "Backstepping" and especially showed that the later has very good stabilizing qualities. In our case, we use the onboard attitude controller provided by Ascending Technologies [5], which is basically a PD-controller.

Although these attitude controllers make it possible to keep the MAVs in a hovering state, there is no possibility to perceive any drift caused by accumulated error. Exteroceptive sensors are thus unavoidable. The most common approach is to mount a DGPS receiver on the MAV. By using a so called Inertial/GPS approach, where data from an IMU and GPS data are fused together, the MAV can be fully stabilized and controlled [6], [7]. Two drawbacks of this approach are the necessity to receive any GPS signal and the lack of precision of the position estimate.

An alternative approach is to use cameras for the localization task. Cameras are lightweight bearing sensors with low power consumption and are relatively cheap to buy. Also, they provide very rich information on the environment. However this vast information has to be processed accordingly. The most simple way is to install a number of external cameras with known location and to have them track the MAV [8], [9], [10]. This method is very efficient for testing purposes and can be used to evaluate other approaches as ground truth reference. However it is not suitable for missions where the installation of an appropriate infrastructure is not feasible.

This approach can also be implemented the other way round: the camera is mounted on the helicopter and tracks a known pattern on the ground [11]. The team of Hamel [12] implemented a visual servoing based trajectory tracking to control an UAV with a mounted camera observing n fixed points. Further methods have also been developed by fusing the visual data with IMU data [13].

The availability of an onboard camera can offer new possibilities. Templeton et al. [14] used a mono vision-based terrain mapping algorithm to estimate the 3D structure of the environment in order to find adequate landing sites (the flight control system still uses GPS data). The problem of autonomously landing a MAV on a known landing platform using vision has been solved already quite early by Saripalli et al. [15]. A vision-based forced landing algorithm has been implemented where a MAV has to localize a good landing area and reach it as fast and safely as possible [16]. Another possibility is to have a MAV tracking a leading MAV with a fixed relative position and orientation. This has been implemented by Chen et al. [17] by constructing an Euclidean homography based on some feature points on the leading vehicle.

Alternatively, stabilizing controllers can be built by means

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 231855 (sFly). Michael Blösch is currently a Master student at the ETH Zurich (email: bloeschm@student.ethz.ch). Stephan Weiss is currently PhD student at the ETH Zurich (email: stephan.weiss@mavt.ethz.ch). Davide Scaramuzza is currently senior researcher and team leader at the ETH Zurich (email: davide.scaramuzza@ieee.org). Roland Siegwart is full professor at the ETH Zurich and head of the Autonomous Systems Lab (email: r.siegwart@ieee.org).

of optical flow considerations [18]. Herisse et al. [19] use an optical flow based PI-controller to stabilize a hovering MAV, they also implemented an automatic landing routine by contemplating the divergent optical flow. Hrabar et al. [20] developed a platform able to navigate through urban canyons. It was based on the analysis of the optical flow on both sides of the vehicle. Also, by having a forward looking stereo camera, they were able to avoid oncoming obstacles.

Based on optical flow some biologically inspired control algorithms have been developed for MAV stabilization [21] [22]. However, the optical flow based pose estimation is also affected by slow drift as it does observe the relative velocity of features only. This counts also for visual odometry based implementations, where the drift is estimated by considering the feature displacements between two successive images [23].

An approach with offboard vehicle tracking equipment was implemented by Ahrens et al. [24]. Based on the visual SLAM algorithm of Davison et al. [25], they build a localization and mapping framework that is able to provide an almost drift-free pose estimation. With that they implemented a very efficient position controller and obstacle avoidance framework. However, due to the simplification they used in their feature tracking algorithm a non-negligible drift persists. Also, they used an external Vicon localization system to control the aerial vehicle with millimeter precision (a system of external cameras that tracks the 3D pose of the vehicle). So far, they did not use the output of the visual SLAM based localization system for controlling the vehicle.

In this paper, we present an approach based on the visual SLAM algorithm of Klein et al. [26]. It enables the MAV 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 or any known pattern in order to obtain a MAV control. The controller is based on a cascaded structure of attitude control and position control. The attitude PD-controller uses the IMU data of the MAV and exhibits a very good performance. The position controller is designed by means of the discrete linear quadratic Gaussian control design with loop transfer recovery (LQG/LTR) 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 the Hummingbird quadrotor from Ascending Technologies [5]. Currently the images are fed via an USB cable to a ground station where the SLAM algorithm is running on. Based on the position estimate the control input are computed and then sent back to the quadrotor. To the best of our knowledge, this is the first implementation of a vision-based MAV controller that can be used in an unknown environment without the aid of any infrastructure based localization system, any beacons, artificial features, or any prior knowledge on the environment. In other words, our platform does not need any external assistance in order to navigate through an unexplored region.

The outline of the paper is as follows: after introducing some notations in section II, we will shortly summarize the SLAM algorithm that has been used here and explain why we chose it for our approach (section III). In section IV we will take a look at the modeling of the system and the parameter identification. After that, we will discuss the controller design (section V) and analyze the entire structure of our approach (section VI). To the end we will have a look at the achieved results and discuss them (section VII).

II. Notations

To facilitate the following considerations we will introduce some notations. We will 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, is 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).
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:

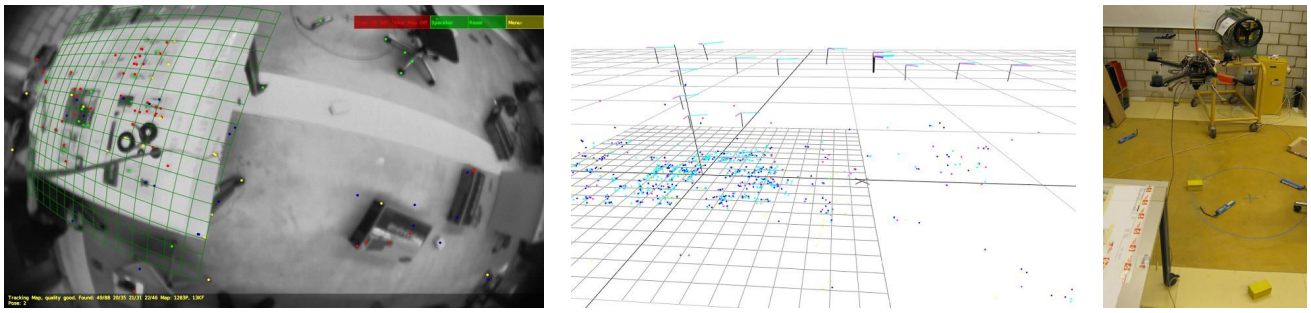
$\boldsymbol{F}_G$:	Gravitational force.
g:	Gravitational acceleration.
m:	Mass of the helicopter.

Please note that we use the Tait-Bryan convention for the Euler decomposition of the rotation matrix R_{HI} into the 3 angles φ, θ and ψ. If the angles represent rotations between two other coordinates frames 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^*).

III. Visual SLAM Based Localization

A. Description of the Visual SLAM algorithm

The presented approach uses the visual SLAM algorithm of Klein et al. [26] in order to localize the MAV from a single camera (see Fig. III). In summary, they split the simultaneous



(a) Camera view (b) SLAM map (c) Hovering quadrotor

Fig. 1. Screenshots of Georg Klein's SLAM algorithm. In (a) the tracking of the FAST corners can be observed, this is used for the localization of the camera. In (b) the 3D map that was build by the mapping thread is shown. The 3-axis coordinate frames represent the location where new Keyframes where added. In (c) the quadrotor in hovering state is represented. Beneath it the mounted camera can be perceived. All three images were taken at the same time.

localization and mapping task into two separately-scheduled threads: the tracking thread and the mapping thread.

The tracking thread is responsible of tracking the selected features in successive frames and computing an estimate of the current camera pose. In this version only FAST corners are tracked and used for the pose estimation.

The Mapping thread uses a subset of all camera images (also called keyframes) to build a 3D point map of the surroundings. The keyframes are selected using some heuristic criteria. After that a batch optimization is applied on the joint state of map points and keyframe poses.

There are several important differences that can be mentioned in comparison to the standard SLAM algorithm of Davison et al. [25]. First of all it does not use any EKF-based state estimation and does not consider any uncertainties, sparing a lot of computational effort. The lack of modeling uncertainities is compensated by using a vast amount of features and the local and global batch optimization. This makes the algorithm fast and the map very accurate.

B. Analysis of the SLAM Algorithm

Splitting the SLAM algorithm in a mapping and a tracking thread brings the advantage that both can run at different speed. The tracker can thus yield fast pose updates while the mapper can use more powerful (slower) map optimization techniques. Compared to frame-by-frame SLAM the mapper does not process every frame. This eliminates to a great extend redundant information processing during slow movements or hovering. These are the main reasons why we choose this SLAM algorithm.

Our downwards looking wide angle camera is always roughly aligned with the z-axis. This ensures large overlapping areas and we can further decrease the frequency at which keyframes are added to the map. In already explored areas, no keyframes will be added and the algorithm's speed remains constant over time while remaining in this area. On the other hand, when exploring new areas the global bundle adjustment can be very expensive, limiting the number of keyframes to a few hundred on our platform (around 50-100 m^2, depending on the keyframe rate).

Another strength of the 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. Also, the algorithm will avoid to add any keyframes in such 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 that the algorithm must initialize new points based on the observations from more than one keyframe. 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 are forced to estimate the map scale by hand and pass it to the controller. We are implementing an online scale estimation aid of an onboard IMU to tackle this issue.

IV. Modeling and Parameter Identification

As our MAV platform we choose the Hummingbird quadrotor from AscTec [5]. The sensors on the platform include 3 gyros, 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 positions coordinates and the yaw angle. In general, the presented method could be implemented on any MAV with sufficiently fast onboard attitude control.

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

The attitude controller controls 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, while the outputs are the

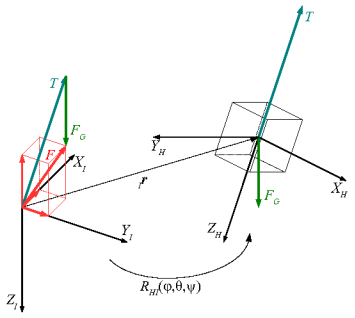


Fig. 2. Point mass model of the helicopter. The inertial frame and the helicopter frame are illustrated. Using the rotation matrix R_{HI} the sum $\boldsymbol{F}$ of the thrust force $\boldsymbol{T}$ and the gravitational force $\boldsymbol{F}_G$ can be projected onto the three axis of the inertial frame.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

estimation of the helicopter position ${}_M\tilde{\boldsymbol{r}}$ and the yaw angle $\tilde{\psi}_{CM}$ computed by decomposing the rotation matrix $\tilde{R}_{CM}$. ${}_M\tilde{\boldsymbol{r}}$ 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 are acting 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 towards the positive z-axis of the inertial coordinate system (see Fig. 2). We can now apply the principle of linear momentum onto the three directions of the inertial coordinate frame. This yields the following matrix equation:

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

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 internal controller to be relatively fast and thus have the following simple relations:

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

A preciser modeling would bring only little improvement to the model's accuracy and would rather increase its complexity. Please note that for equation (1) it is important that the z-axis of the inertial frame points toward the center of gravity. Theoretically, the principle of linear momentum could also be applied in the coordinate system of the SLAM map M (which is also an inertial frame), but due to a slight orientation drift of the map it was necessary to include an additional inertial frame I and adapt the corresponding orientation R_{MI} (more in section VI).

The attitude controller dynamics from the inputs φ^* and θ^* to the angles φ and θ are fast. We model them as two separated second-order systems with the 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 Fig. 3). This yields a value

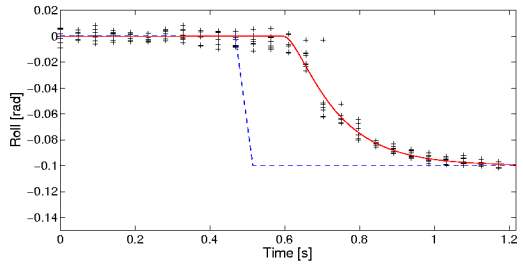


Fig. 3. Measured step response on the pitch input of the quadrotor device. The system is modeled as a second-order system with time delay. This yields a critical frequency ω of 15.92 rad/s, a damping d of 1.22 and a time delay T_d of 80.6 ms (for image transmission via USB cable).

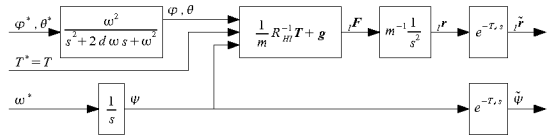


Fig. 4. Model of the entire system with the roll angle φ^*, the pitch angle θ^*, the total thrust T^* and the yaw rate ω^* as inputs. The outputs are the position of the helicopter ${}_I\tilde{\boldsymbol{r}}$ and the yaw angle $\tilde{\psi}$. The attitude controller dynamics and the time delay are included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

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 data transmission and SLAM algorithm, so that its value is the same for all outputs. Depending on the data transmission method the delay varies between 80.6 ms (USB cable) and 250 ms (Wi-Fi n-standard).

All in all the system can be represented like in Fig. 4. Please note that for the sake of simplicity we do not model any disturbances or noise.

V. Controller Design

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

$$ \begin{pmatrix} {}_IF_x \\ {}_IF_y \\ {}_IF_z \end{pmatrix} = -T \begin{pmatrix} \cos\psi \sin\theta \cos\varphi + \sin\psi \sin\varphi \\ \sin\psi \sin\theta \cos\varphi - \cos\psi \sin\varphi \\ \cos\theta \cos\varphi \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix} $$

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

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

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

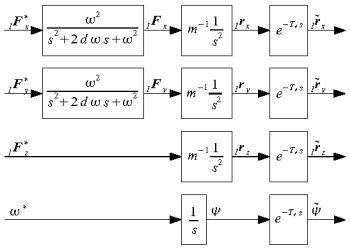


Fig. 5. The four decoupled linear systems. They were obtained by transforming the system inputs of Fig. 4 and exchanging the order of the nonlinear input transformation and the second order system blocks. This can be justified by the smoothness of the nonlinear transformation and the high speed of the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

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

Assuming that the attitude control's dynamic is fast and smooth enough, the second-order system block (attitude controller dynamics) and the control input transformations 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 Fig. 5).

For the yaw control we simply apply a constant velocity ω^* when the angular error exceeds a certain value.

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 15-30Hz). For the Nyquist frequency we take 7.5 Hz (half of the minimal measurement frequency). During the tuning, this represents the main limitation of the control performance.

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. Supplementary, due to varying battery power and tilt angles calibrations some integrating action has to be introduced in all position coordinates by expanding the system with an output error integrating part. Now the corresponding system matrices F, G, C and D can be computed. Applying the LQG/LTR procedure with feed-forward action yields the structure in Fig. 6.

The resulting closed loop system has its poles like in Fig. 7. Except for the four poles induced by the time delay, all poles are between 1-30 rad/s. This enables the system to correct an initial error of 1 m with a T_{90} time of around 1 seconds and 20% overshoot (Fig. 8). At the expense of the performance, we attempt to maximize the robustness of the system in order to handle the modeling errors and external

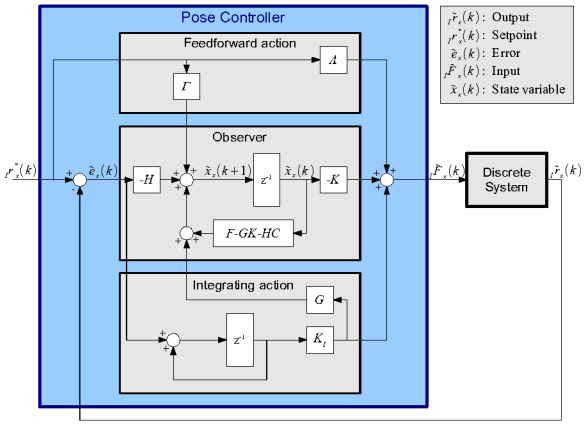


Fig. 6. LQG/LTR controller structure for the position control in x. The structure for y and z are analogous. It results from the combination of a state observer and a state feedback controller. An integrating part and feedforward action were included in order to obtain unity steady state gain and to 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.

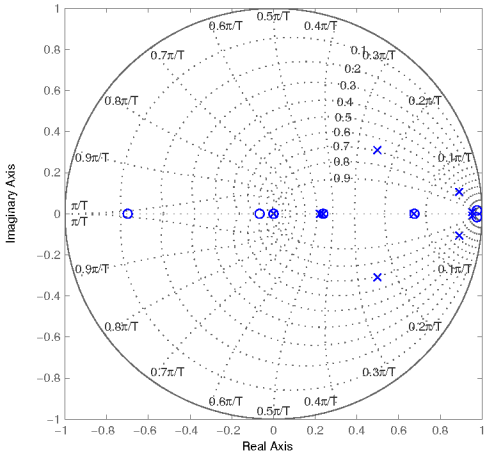


Fig. 7. Pole-zero map of the closed-loop system for x or y. Except for the four poles induced by the time delay all poles lie within 1-30 rad/s. No pole has a damping lower than 0.5. A zero situated at -7.2 is not displayed on the plot.

disturbance. However, to attain fast error correction at the beginning of the flight a sufficient large integrating part has to be maintained. Examining the Nyquist plot (Fig. 9) we can observe a phase margin of 27.7 degrees and a gain margin of 5.5 dB, suggesting an acceptable robustness.

The main advantages of the LQG/LTR controller design is that it filters the measurements and estimates the velocity of the helicopter. This is especially useful to handle discontinuities of the SLAM pose estimate caused by the bundle adjustment procedure or wrong data association. Also compared to the standard PID approach it enables us to handle the non-negligible time-delay in order to obtain faster controllers. The additional loop transfer recovery allows us

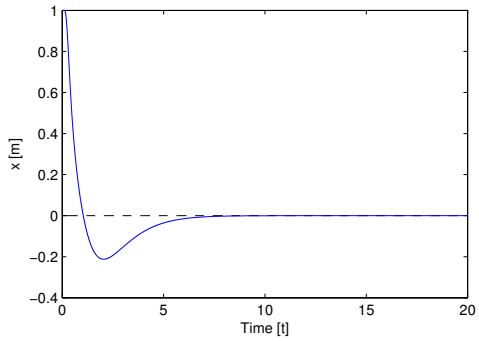


Fig. 8. Time domain system response to an initial error of 1 m in x or y. The controller is able to correct the error with a T_{90} time of around 1 seconds and an overshoot of 20%. The performance is limited by the relatively slow measurement rate and the time delay of the system.

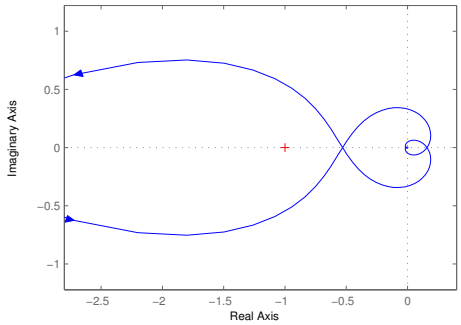


Fig. 9. Nyquist plot of the system (open loop). Phase margin: 27.7 degrees. Gain margin: 5.5 dB.

to find a good trade-off between robustness and performance.

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

VI. Final System Structure, Final Implementation

We use the Hummingbird quadrotor platform from Ascending Technologies [5]. A high performance onboard controller enables the stabilization of both tilt angles and the yaw rate at desired reference values sent via an XBee radio. Beneath the quadrotor a 12g USB uEye UI-122xLE is installed which gathers 752x480 images with global shutter. At the moment the images are transmitted through an USB cable linked to the ground station. The computations on the ground stations are done on a Intel Core 2 CPU 2x2GHz processor. All code is implemented in C++.

In the flow diagram (see Fig. 10) the entire closed-loop system is represented. The SLAM algorithm and the controller are both implemented on the ground station.

As the vision based localization does not work when the helicopter is landed (the camera is too near to the floor), the take off is only feasible if the initial land-patch beneath it is already stored in the map. Giving increasing thrust the helicopter can then fly blindly until it re-finds the map and

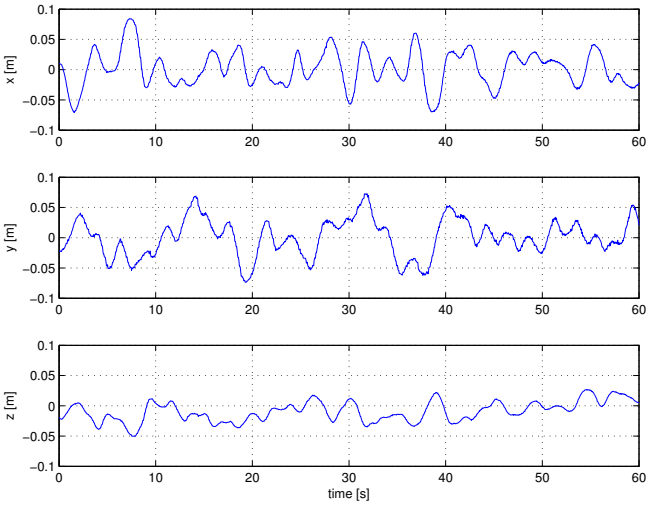


Fig. 11. Position error in the x,y and z positions. The value remains between $\pm$ 10 cm. The z position is more accurate than the x and y positions.

stabilizes itself (the position can be tracked from a height of ca 15 cm). We are currently including an algorithm that is able to take off from ground over a known pattern and initialize the map autonomously.

We observed stability problems arising from the scale and orientation drift of the SLAM map. Due to the projective nature of a single camera the scale of the map can diverge from it's original value and lead the quadrotor to crash. Currently the scale can be adjusted manually, however we are working on a framework where an online scale estimation algorithm is included. The robustness of the designed controller allows the system to handle a relative scale error of around $\pm 20\%$.

The rotational drift in the map does also lead to instability if not considered. At the moment this problem is solved by automatically re-aligning the inertial coordinate frame every 40 cm. For that 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). When 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). In order to retain a smooth position estimates an offset on the SLAM position is introduced and adapted at each re-alignment. The entire procedure limits the progressive speed of the MAV and leaves the mapping thread of the SLAM algorithm some time to expand the map.

VII. Results and Discussion

In Fig. 12 the flight path of 60 seconds hovering can be seen. Note that during hovering no keyframes have to be added and the SLAM algorithm can focus on position tracking. The position error has an RMS value of 2.89 cm in x, 3.02 cm in y and 1.86 cm in z, what yields an absolute error value RMS of 4.61 (see Fig. 11 and Fig. 12).

The platform is also able to fly to desired setpoints. For that the path is split into waypoints. The distance between them is chosen so that the helicopter can re-align the orientation of the inertial coordinate frame at each waypoint. Here,

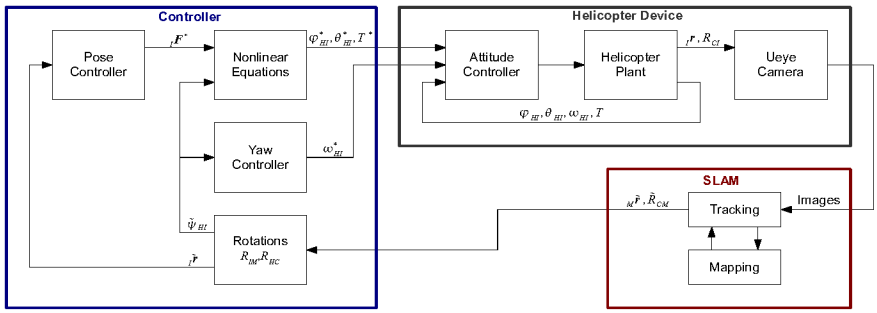


Fig. 10. Closed-loop controller structure. The entire data flow is represented. Receiving the images from the mounted camera the SLAM algorithm computes the estimate of the pose of the camera. This is transformed into the position and yaw angle of the helicopter in the inertial frame and fed to the controller. After the nonlinear control input transformations (Eq. (4),(5),(6)) the reference values are sent back to the attitude controller on the quadrotor.

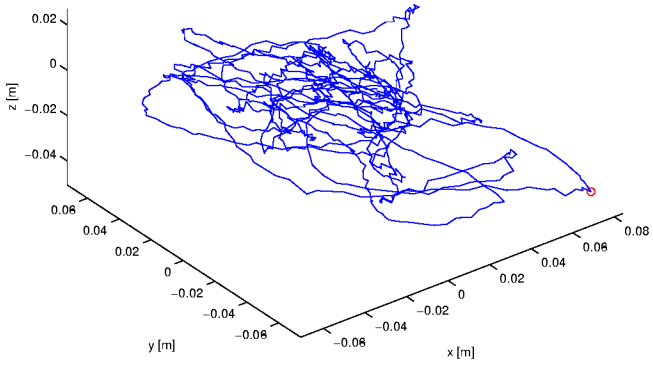


Fig. 12. Position error while hovering during 60 seconds. The RMS value of the position error is 2.89 cm in x, 3.02 cm in y and 1.86 cm in z. The maximum runaway has an absolute error value of 11.15 cm (marked with a red o).

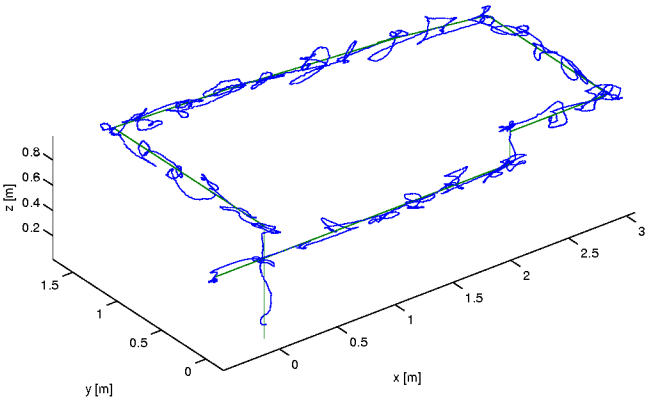


Fig. 13. Path that the helicopter has flown. This does not represent ground truth, it is the pose estimation of the SLAM algorithm. However the attitude of the helicopter can be observed while successfully flying a rectangular loop and landing on the ground. The RMS value of the position error is 9.95 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 a region of 3.5x2x1m^3

a little higher RMS value is obtained as in the hovering mode (see Fig. 13). The map consisted of 7 keyframes at the start. While expanding the map the SLAM algorithm processes more than 40 additional keyframes. At each waypoint the helicopter stabilizes itself and waits until the RMS value is small enough (10 cm in absolute error value) to re-align the map. We observed that the systems remains stable even in the case where the re-alignment is imprecise.

The localization is also very reliable on non-flat terrain. During the trajectory in Fig. 13 the quadrotor flies pass the edge of a table it has started on. More problematic could be the scenario where the quadrotor would need to fly very near the ground or an obtrusive object (like passing over a wall). In these cases the scene is passing relatively fast within the camera frame and the flight speed would have to be limited. Adding supplementary camera looking in other directions could be of benefit in such scenarios.

In Fig. 14, we can see the map that was built during the flight. It is composed of 52 keyframes and 4635 map points. It represents approximately a surface of 15 m^2. Even

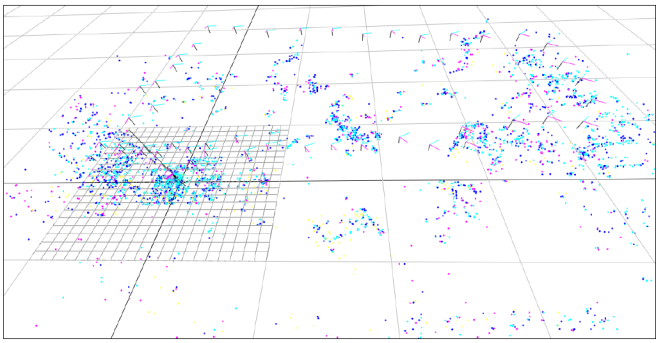


Fig. 14. 3D view of the built map. It contains 4635 map points, observed in 52 different keyframes.

though there might be some scale and orientation drift within the map, the system remains stable. Locally, the quadrotor can localize itself correctly and thus the control quality is maintained.

Some failure modes have also to be mentioned. Due to the lack of features or to varying illumination it is possible

that the tracker cannot find enough features disabling the localization. In big maps the mapping thread consumes too much calculation power for new keyframes, preventing the tracking thread from updates.

The achieved results show that our platform can autonomously fly through a larger unknown indoor environment with high accuracy. The system is robust against external disturbances and can handle modeling errors. Some outdoor tests confirm the controller's robustness, which was able to handle quit strong changing winds.

VIII. Conclusion

This paper presented a vision based MAV control approach. The pose was estimated by means of the visual SLAM algorithm of Klein et al. 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 within an indoor laboratory. For that 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. Apart from some minor map drift the vehicle can control its position up to a few centimeters of error (RMS around 2-4 cm). We successfully built an autonomous MAV platform which is able to navigate in an unknown and unstructured environment in a very robust and accurate way.

IX. Acknowledgements

The authors would like to thank Gabriel Nützi and Daniel Eberli for their critical and fruitful discussions and their very valuable contributions.

References

[1] P. Castillo, R. Lozano, and A. Dzul, "Stabilization of a mini-rotorcraft having four rotors," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, vol. 3, 2004, pp. 2693–2698 vol.3.

[2] P. Kemao, D. Miaobo, M. Chen Ben, C. Guowei, L. K. Yew, and T. H. Lee, "Design and implementation of a fully autonomous flight control system for a uav helicopter," in *Proc. Chinese Control Conference CCC 2007*, 2007, pp. 662–667.

[3] G. Cai, A. K. Cai, B. M. Chen, and T. H. Lee, "Construction, modeling and control of a mini autonomous uav helicopter," in *Proc. IEEE International Conference on Automation and Logistics ICAL 2008*, 2008, pp. 449–454.

[4] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proc. IEEE International Conference on Robotics and Automation ICRA 2005*, 2005, pp. 2247–2252.

[5] *AscTec Hummingbird Quadrotor Helicopter*, Ascending Technologies GmbH, http://www.asctec.de.

[6] B. Yun, K. Peng, and B. M. Chen, "Enhancement of gps signals for automatic control of a uav helicopter system," in *Proc. IEEE International Conference on Control and Automation ICCA 2007*, 2007, pp. 1185–1189.

[7] N. Abdelkrim, N. Aouf, A. Tsourdos, and B. White, "Robust nonlinear filtering for ins/gps uav localization," in *Proc. 16th Mediterranean Conference on Control and Automation*, 2008, pp. 695–702.

[8] E. Altug, J. P. Ostrowski, and R. Mahony, "Control of a quadrotor helicopter using visual feedback," in *Proc. IEEE International Conference on Robotics and Automation ICRA '02*, vol. 1, 2002, pp. 72–77 vol.1.

[9] S. Park, D. H. Won, M. S. Kang, T. J. Kim, H. G. Lee, and S. J. Kwon, "Ric (robust internal-loop compensator) based flight control of a quad-rotor type uav," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 3542–3547.

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

[11] B. Ludington, E. Johnson, and G. Vachtsevanos, "Augmenting uav autonomy," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 63–71, 2006.

[12] T. Hamel, R. Mahony, and A. Chriette, "Visual servo trajectory tracking for a four rotor vtol aerial vehicle," in *Proc. IEEE International Conference on Robotics and Automation ICRA '02*, vol. 3, 2002, pp. 2781–2786.

[13] T. Cheviron, T. Hamel, R. Mahony, and G. Baldwin, "Robust nonlinear fusion of inertial and visual data for position, velocity and attitude estimation of uav," in *Proc. IEEE International Conference on Robotics and Automation*, 2007, pp. 2010–2016.

[14] T. Templeton, D. H. Shim, C. Geyer, and S. S. Sastry, "Autonomous vision-based landing and terrain mapping using an mpc-controlled unmanned rotorcraft," in *Proc. IEEE International Conference on Robotics and Automation*, 2007, pp. 1349–1356.

[15] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Vision-based autonomous landing of an unmanned aerial vehicle," in *Proc. IEEE International Conference on Robotics and Automation ICRA '02*, vol. 3, 2002, pp. 2799–2804.

[16] L. Mejias, P. Campoy, K. Usher, J. Roberts, and P. Corke, "Two seconds to touchdown - vision-based controlled forced landing," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 3527–3532.

[17] J. Chen and D. M. Dawson, "Uav tracking with a monocular camera," in *Proc. 45th IEEE Conference on Decision and Control*, 2006, pp. 3873–3878.

[18] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "Mav obstacle avoidance using optical flow," in *Proc. IEEE International Conference on Robotics and Automation ICRA 2010*, 2010.

[19] B. Herisse, F.-X. Russotto, T. Hamel, and R. Mahony, "Hovering flight and vertical landing control of a vtol unmanned aerial vehicle using optical flow," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2008*, 2008, pp. 801–806.

[20] S. Hrabar, G. S. Sukhatme, P. Corke, K. Usher, and J. Roberts, "Combined optic-flow and stereo-based navigation of urban canyons for a uav," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 3309–3316.

[21] F. Ruffier and N. Franceschini, "Visually guided micro-aerial vehicle: automatic take off, terrain following, landing and wind reaction," in *Proc. IEEE International Conference on Robotics and Automation ICRA '04*, vol. 3, 2004, pp. 2339–2346 Vol.3.

[22] J.-C. Zufferey and D. Floreano, "Fly-inspired visual steering of an ultralight indoor aircraft," *IEEE Trans. Robot.*, vol. 22, no. 1, pp. 137–146, 2006.

[23] S. G. Fowers, D.-J. Lee, B. J. Tippetts, K. D. Lillywhite, A. W. Dennis, and J. K. Archibald, "Vision aided stabilization and the development of a quad-rotor micro uav," in *Proc. International Symposium on Computational Intelligence in Robotics and Automation CIRA 2007*, 2007, pp. 143–148.

[24] S. Ahrens, D. Levine, G. Andrews, and J. P. How, "Vision-based guidance and control of a hovering vehicle in unknown, gps-denied environments," in *Proc. IEEE International Conference on Robotics and Automation ICRA '09*, 2009, pp. 2643–2648.

[25] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007.

[26] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality ISMAR 2007*, 2007, pp. 225–234.