2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Stereo Vision based indoor/outdoor Navigation for Flying Robots

Korbinian Schmid[1], Teodor Tomić[2], Felix Ruess[1], Heiko Hirschmüller[1] and Michael Suppa[1]

***Abstract*— We introduce our new quadrotor platform for realizing autonomous navigation in unknown indoor/outdoor environments. Autonomous waypoint navigation, obstacle avoidance and flight control is implemented on-board. The system does not require a special environment, artificial markers or an external reference system. We developed a monolithic, mechanically damped perception unit which is equipped with a stereo camera pair, an Inertial Measurement Unit (IMU), two processor-and an FPGA board. Stereo images are processed on the FPGA by the Semi-Global Matching algorithm. Keyframe-based stereo odometry is fused with IMU data compensating for time delays that are induced by the vision pipeline. The system state estimate is used for control and on-board 3D mapping. An operator can set waypoints in the map, while the quadrotor autonomously plans its path avoiding obstacles. We show experiments with the quadrotor flying from inside a building to the outside and vice versa, traversing a window and a door respectively. A video of the experiments is part of this work. To the best of our knowledge, this is the first autonomously flying system with complete on-board processing that performs waypoint navigation with obstacle avoidance in geometrically unconstrained, complex indoor/outdoor environments.**

I. Introduction

A. Motivation

Robots can be valuable helpers in search and rescue (SAR) and disaster management scenarios. Prof. Hajime Asama showed impressively in his 2012 IROS plenary lecture how robots were used to analyze, survey and partly clean up the area around the Fukushima Daiichi nuclear power plant after the meltdown caused by the catastrophic earthquake and tsunami in 2011. Nevertheless, there is still a gap between robotic research and real applications of mobile robots in SAR scenarios, which is a strong motivation for our work.

Using mobile robots in real disaster management scenarios requires the system to provide a certain degree of autonomy. In many situations a stable, high bandwidth radio link between the robot and a ground station can not be guaranteed. Furthermore, external navigation aids such as GPS are unreliable in areas such as urban canyons, or not available at all inside buildings. Therefore, at least sensor data needed for safe navigation has to be processed on-board the system.

Especially for flying robots, this requirement imposes a big challenge, as the payload of these systems is usually strongly limited. Payload limitations accompany limitations

[1]German Aerospace Center (DLR), Robotics and Mechatronics Center (RMC), Department of Perception and Cognition Münchner Str. 20, 82234 Wessling, Germany

[2]German Aerospace Center (DLR), Robotics and Mechatronics Center (RMC), Department of Mechatronic Components and Systems, Münchner Str. 20, 82234 Wessling, Germany

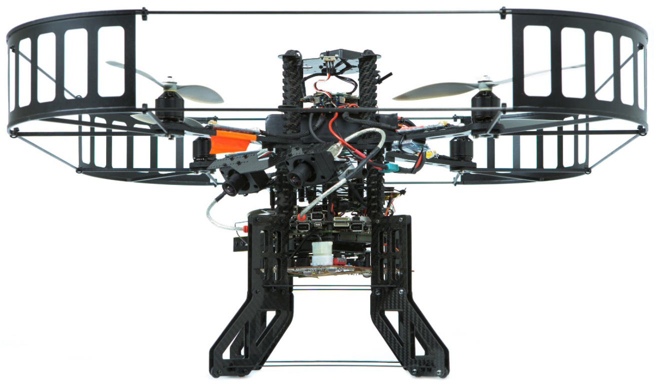


Fig. 1. Experimental quadrotor platform.

in computational resources. Sensor data processing and navigation algorithms have to be robust, efficient and fast. In contrast to ground robots, flying robots like quadrotors are inherently unstable and have to be actively controlled at any time. Considerable measurement time delays can be induced, as the sensor data is processed on board resource-limited systems. These delays have to be compensated for control applications, as flying systems are highly agile.

The choice of sensors is crucial for navigation, surveillance and planning abilities of the robot. Data from 3D depth sensors is well suited for these tasks. Laser scanners provide reliable measurements but are rather heavy for flying robots. Sensors like Kinect are small and lightweight but work only indoors. Time-of-flight cameras also work outdoors but have a limited resolution. Stereo cameras are lightweight and provide a high resolution. Nevertheless, high resolution stereo processing on resource-limited systems is a challenging task.

In our previous work [1] we analyzed the influence of measurement time delays and frequency on the quality of state estimation for highly dynamic flying systems. The results motivated us to use stereo vision as the only exteroceptive sensor for navigation. We designed a navigation box [2] including an IMU, a stereo-camera pair, a realtime system for sensor data fusion and control, a non-realtime system for ego-motion calculation and navigation, and an FPGA board for stereo image processing. We optimized the system for weight and integrated it into our new quadrotor platform (Figure 1), which is an advancement of our previous system [3]. Stereo depth images with 0.5 Mpixel resolution are calculated on board the quadrotor platform at 14.6 Hz using an FPGA implementation of the Semi Global Matching (SGM) algorithm [4]. Our sensor data fusion framework compensates for time delays of about 250 ms in the calculated stereo ego-motion and provides an estimate of the current full 3D robot pose, velocity and IMU sensor biases. All states are locally drift-free due to our keyframe-based stereo odometry. The estimates are used for quadrotor control and on-board 3D mapping using stereo depth images only. We realized on-board 2D path planning and collision avoidance by projecting 3D map obstacles at the height of the quadrotor to the ground plane. The system performance is demonstrated in a challenging indoor/outdoor flight. The quadrotor is commanded from inside a building corridor through a window to the outside, around the building, back through a door to the inside and following the corridor to the starting point. An operator only chooses some way points in the incrementally built on-board 2D obstacle map. The quadrotor then plans its path autonomously. Newly appearing obstacles are avoided by continuous, automatic re-planning.

B. *Related Work*

Flying robots capable of autonomous navigation in GPS-denied environments have become an area of increasing research interest. Bachrach et al. [5] use a laser scanner for pose estimation of Micro Aerial Vehicles (MAV). An off-board SLAM algorithm using laser scan matching performs pose-graph optimizations and loop detection. With off-board planning, autonomous exploration and a flight through a window was shown. Shen et al. [6] also utilize a laser scanner for multi-floor navigation, but perform all computations including SLAM with loop closure on-board the MAV. Based on his previous work, Shen et al. [7] added an RGB-D sensor. With a focus on frontier exploration and map representation, autonomous operation in a large multi-floor indoor environment was shown. The MAV pose is however still estimated via laser scan matching, limiting it to a 2.5D environment with vertical walls. Outdoor navigation is only partially possible as long as multiple walls are still seen by the laser scanner.

Huang et al. [8] fuse visual odometry estimates from an RGB-D sensor with IMU measurements in an EKF for local navigation. Pose and velocity estimation for real-time control of the MAV is performed on-board. The RGB-D data is also transmitted to an off-board laptop for global navigation and loop closure with a SLAM algorithm. Due to significant processing delay, a state history needs to be kept. Upon a new SLAM correction the state in the history is modified and all future estimates recomputed. Usage of the RGB-D sensor such as Kinect also does not permit outdoor operation in direct sunlight.

Heng et al. [9] use stereo cameras for on-board mapping and path planning to avoid obstacles. Pose estimation is done using known artificial markers or with an external Vicon motion tracking system.

Our system is most closely related to the work of Fraundorfer et al. [10]. A front-looking stereo camera is used as the main sensor to build a global 3D occupancy map with 0.1 m resolution on-board the MAV. A 2D slice (at a fixed height) of the 3D Octomap is used for on-board planning and obstacle avoidance as well as frontier-based exploration and wall following. The state estimation is split into several parts. Using the stereo cameras, visual odometry is computed at 10 Hz. A reference frame is maintained as long as feasible to avoid local drift. A downward looking optical flow camera (in conjunction with a sonar for altitude) provides velocity measurements, which are used in a simple Kalman Filter to estimate the partial state. The integrated velocities are then combined with the output of the visual odometry via a low-pass filter for providing a complete pose estimate. While this works well for their stated assumptions, it can become problematic in a real SAR scenario where the floor is not planar or the small roll and pitch angle approximation is violated in dynamic flight. For example the usage of sonar and optical flow as speed estimate for the controller does not allow for a flight through a window. Furthermore, the pose-graph SLAM algorithm is run off-board and only used for post-processing.

Another notable work for stereo image processing has been shown by Honegger et al. [11]. A 376×240 pixel depth image and an optic flow field are computed at 127 Hz on an FPGA. In contrast to optic flow with high frame rates, our FPGA implementation aims for high resolution, high quality depth images. Nevertheless, our lower frame rate of 14.6 Hz is sufficient for an agile system with fast movements, due to our visual odometry approach that can reliably and accurately compute the ego-motion even if successive images overlap by just 50%.

In contrast to the previous work and to the best of our knowledge, we present the first flying system that can navigate autonomously in complex, geometrically unconstrained indoor/outdoor environments without relying on vertical walls or a flat ground while all data is processed on-board.

II. System Design

The hardware design of our system, introduced in the following section, is carefully adapted to the needs of the software components introduced in section II-B.

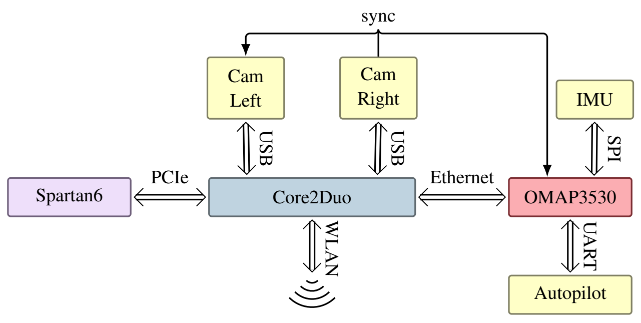


Fig. 2. Hardware block diagram of the system components.

A. *Hardware*

Our experimental quadrotor platform, shown in Figure 1, has the dimensions of 0.60 m $\times$ 0.60 m $\times$ 0.35 m, a maximum horizontal diameter of 0.77 m and a weight of

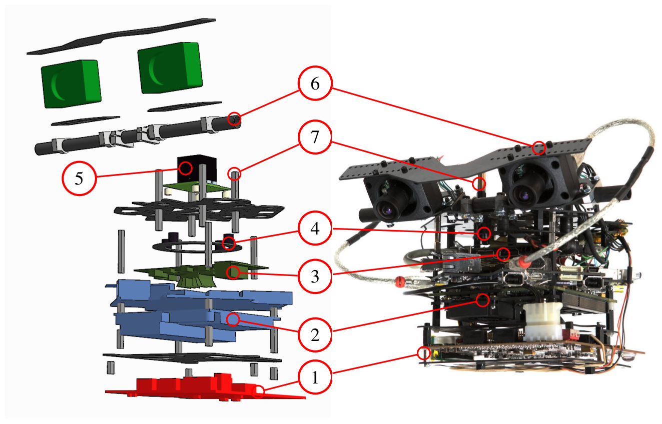


Fig. 3. Exploded and assembled view of the processing stack, showing (1) FPGA card, (2) Core2Duo board, (3) Gumstix (4) plate that carries the stack, (5) IMU (not visible in assembled view) and (6) stereo camera assembly. The stack is mounted to the quadrotor via (7), which is rigidly fastened to the plate (4). Dampers are placed on (4), so that the whole stack is damped w.r.t. the quadrotor frame. This allows fast disassembly from the quadrotor frame without losing IMU to camera calibration.

1.63 kg (excluding batteries). The integrated navigation box is shown in Figure 3.

1) Navigation Box Design: The design of the navigation box takes several key requirements into account. Firstly, it is decoupled of the quadrotor frame, so that all sensors are integrated with the processing electronics. The only physical connection to the quadrotor frame is via four screws at point (7) in Figure 3. In this way, integration, calibration and testing can be done without the quadrotor. Secondly, due to the vibrations induced by the rotors, the navigation box must to be damped to cut off some of the high frequency vibrations measured by the IMU. Therefore, we added an extra plate (4) with dampers that connects to the quadrotor frame. The navigation box is fastened on top of the dampers on this plate. In this way, the whole mass of the navigation unit is used to lower the natural frequency of vibrations acting on the box. Hence, the dampers and mass of the unit act as a mechanical lowpass filter. Thirdly, the sensors used for state estimation – IMU and cameras – have to be rigidly connected. Therefore, they are mounted on the same plate, which is additionally stiffened. Finally, the camera stereo pair is designed to be stiff in all rotational directions. All custom components are milled out of carbon fiber plates or aluminum (camera to rod connection).

2) Navigation Box Electronic Components: The hardware configuration of the integrated navigation box is similar to the hand-held pose estimation device that we developed earlier [2], but optimized for weight. We use a stereo camera pair consisting of two hardware synchronized Point Grey Firefly cameras connected via USB to a Core2Duo SU9300@1.86GHz processor board. The cameras are equipped with light-weight *Computar HM0320KP* lenses ($f = 3$ mm, horizontal FOV $= 80.5°$, manual aperture). Stereo data processing is done on a Spartan 6 LX75 FPGA Eval Board which is connected via PCI Express. Figure 2 depicts the block diagram of the system.

Additionally to the x86 platform, we use a Gumstix computer board providing a OMAP3530 ARM Cortex A8@720MHz processor. An ADIS16407 Inertial Measurement Unit (IMU) integrating a triaxial digital accelerometer, gyroscope, magnetometer and a barometer is connected via SPI. The hardware trigger of the stereo camera pair is registered at the Gumstix computer board.

We run Linux on both computer platforms. The communication between the boards is carried out via Ethernet. A software wireless LAN/Ethernet bridge on the Core2Duo board integrates the on-board network transparently into the ground station network. Furthermore, we synchronize system clocks via Precision Time Protocol V2.

An overview of the weight of the individual components which is crucial for all MAVs is summarized in Table I.

TABLE I
WEIGHT OF SYSTEM COMPONENTS

Component	Weight
FPGA board	95 g
Core2Duo stack	345 g
Gumstix board	37 g
IMU incl. baseboard	22 g
Cameras incl. lenses	2 x 33 g
Mount and cables	171 g
Total	739 g

B. Software

The software architecture of our system is depicted in Figure 4. We use ROS (Robot Operating System) as middleware connecting all software components. Modules with hard realtime requirements implement their own message queues using the linux FIFO scheduler.

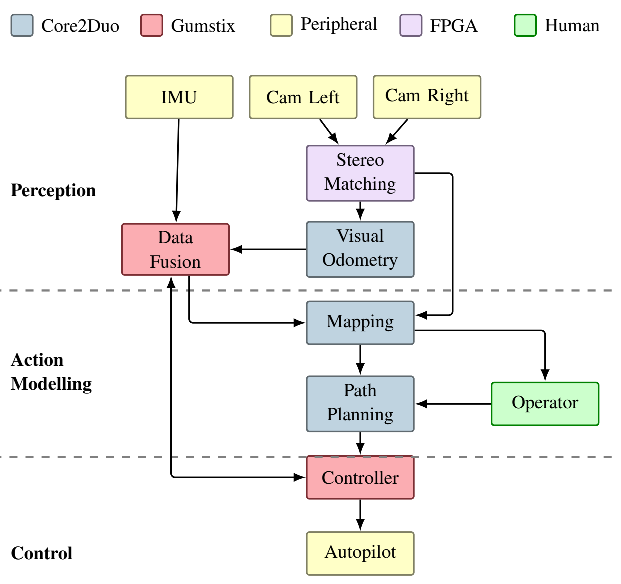


Fig. 4. Block diagram of software components.

1) Stereo Matching and Visual Odometry: The image processing software is responsible for computing dense depth images from stereo images and the visual odometry from subsequent camera images.

The depth image is computed from rectified stereo images by Semi-Global Matching (SGM) [4]. The method performs pixelwise matching, supported by a global cost function that prefers piecewise smooth surfaces. The Hamming distance of Census transformed images is used as matching cost [12], since it offers a high radiometric robustness that is needed for processing real images [13]. SGM delivers dense, high quality depth images with high spatial resolution (i.e. fine structures are visible in the depth image). The method is not sensitive to the choice of parameters, which makes parameter tuning needless in practice.

We use a Spartan 6 FPGA implementation of SGM that is an optimized version of Gehrig et al. [14]. The implementation processes rectified stereo images in a resolution of 1024×508 pixels with 12 bit radiometric depth and 128 pixel disparity range in 68 ms.

The depth image is used for obstacle avoidance and mapping, but it also serves as base for visual odometry, which works on subsequent left camera images. The used method [15], [16] has been developed for fast movements and rather low framerates. It performs corner detection and tries to find initial correspondences by correlating the Rank [12] signature of all corners against each other. Outlier detection is based on relative distances of corner pairs that are reconstructed in 3D by using the dense depth image of SGM. Due to using 3D features, the motion can be calculated with six degrees of freedom with a theoretical minimum of three correspondences. In contrast, mono camera based approaches can only determine five degrees of freedom (i.e. no scale) from a minimum of five correspondence, which requires more complexity for outlier detection and is therefore potentially slower and less robust in practice.

The incremental visual odometry method has been extended by using keyframes and estimating the motion error as well [17]. Keyframes are used by storing old images into a small, fixed sized list of previous images. The motion to a new image is always determined from all previous images in the list. The motion that minimizes the estimated overall motion error is used as resulting motion and the new image replaces the one in the list with the highest overall error. This mechanism reduces an erroneous motion drift for a slow moving system and is drift free for a system that is standing still or moving on the spot.

In our implementation, depth images and the visual odometry are computed at 14.6 Hz with a latency between 223 ms to 288 ms. Everything except SGM runs on the Core2Duo board with a CPU load of 129 % [2].

2) Data Fusion: The visual odometry is fused with IMU data for getting a system state estimate that is used for mapping and control. The requirements for mapping are rather relaxed: the stereo camera pose with respect to the mapping frame at the time of image acquisition is needed. This pose could also be calculated from visual odometry only, but fusion estimates are more robust against vision dropouts as was shown in our previous work [2].

By contrast, system state estimation for control of flying robots has to fulfill some requirements – the fast dynamics of quadrotors have to be reflected in a high controller bandwidth. Therefore, the system state has to be available at a high rate, at least at the rate of the lowest controller cascade. Furthermore, the controller needs the system state at the current time without any delays coming from sensor data processing. In our case, the stereo odometry system introduces a measurement time delay of about 250 ms.

Our filter framework shown in Figure 5 considers these requirements – the system state estimate is available at a rate of up to 819 Hz, i.e. the full IMU data rate, while measurement time delays are compensated. As discussed above, the position estimation accuracy is increased by processing key frame odometry instead of simple incremental odometry [2]. In the following we will summarize the structure of the filter framework [1].

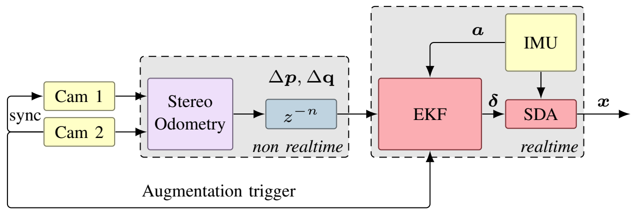


Fig. 5. State estimation system design.

We define the direct system state as

$$x = \begin{pmatrix} p_{nb}^{n,T} & v_{nb}^{n,T} & q_b^{n,T} & b_a^{b,T} & b_\omega^{b,T} \end{pmatrix}^T \quad (1)$$

defining body position and velocity in the navigation frame, the orientation quaternion between the body and navigation frame and the IMU accelerometer and gyroscope biases. The navigation frame is defined with the z-axis pointing downwards and the x-and y-axes coinciding with the corresponding body axes at the starting point. x is calculated at 200 Hz by the computationally inexpensive Strap Down Algorithm (SDA) using accelerometer and gyroscope measurements from the IMU.

The direct system state will accumulate errors due to integration of noisy measurements and linearization effects in the SDA. Therefore, we estimate the errors defined as the indirect system state of an Extended Kalman Filter (EKF):

$$\delta = \begin{pmatrix} \delta_p^{n,T} & \delta_v^{n,T} & \delta_\psi^{n,T} & \delta_{b_a}^{b,T} & \delta_{b_\omega}^{b,T} \end{pmatrix}^T \quad (2)$$

The position, velocity, orientation and bias errors, respectively, are estimated in every filter step and used to correct the direct system state. By using the indirect filter formulation in feedback configuration, the computationally expensive calculation of system state errors can be executed at a much lower rate than the direct system state calculation.

For error estimation, corresponding to the EKF update step, we use two measurement sources. Most of the time,

(a) On-board left camera view.

(b) On-board calculated Semi Global Matching (SGM) depth image.

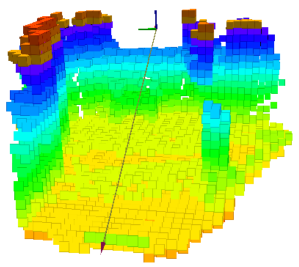

(c) On-board calculated 3D map. The coordinate frame shows the current pose of the quadrotor.

Fig. 6. Quadrotor crossing a window from an indoor corridor to the outside.

the IMU acceleration measurement is dominated by the gravity vector. We fuse this pseudo gravity measurement for roll and pitch angle stabilization, which is especially important for flying platforms. The second measurement update is provided by the stereo odometry system in form of a delta position ($\boldsymbol{\Delta p}$) and delta orientation quaternion ($\boldsymbol{\Delta q}$) between the delta measurement start at time t_s and the end at time t_e. For a time delayed measurement arriving at t_n we know $t_s^n < t_e^n \leq t_n$. To process these time delayed delta measurements we clone the system error pose into the filter error state and save the direct system state at the time of a camera hardware trigger. Only the part of the error state vector reflecting the current state needs to be propagated in the Kalman Filter propagation step, while the augmented states stay constant. At the arrival of a time delayed delta pose measurement the augmented states are referenced in the measurement matrix and hence the measurement delay is compensated implicitly. The calculated correction improves all system states including the augmented system states in the past. Augmentations that will not be needed for future measurements are removed from the filter.

3) Control: We employ a cascaded controller structure, which consists of a low level PD attitude controller (running on the Autopilot at 1 kHz), and a position controller, which runs on the Gumstix at 50 Hz. The PID position controller sends attitude commands via a UART connection.

The trajectory is generated by linearly interpolating the position between waypoints and lowpass filtering the result. In this way, a smooth velocity and acceleration feedforward signal is generated. We advance to the next waypoint when the quadrotor is within the specified tolerance of the current waypoint.

4) Mapping and Path Planning: A 3D occupancy map is computed on-board using the Octomap Library [18]. The SGM stereo depth image calculated by the FPGA is converted to a point cloud and downsampled with a Point Cloud Library (PCL) grid filter. The resulting thinned out cloud is inserted into the Octomap using the best current pose estimate from the data fusion.

For collision avoidance and path planning, a horizontal layer at the altitude of the quadrotor and its height as thickness is cut out of the 3D map. The occupied cells are projected down on the horizontal plane to get a 2D collision map. The ROS Navigation package is used to generate a collision-free path to a waypoint given by the operator. In the first step, obstacles in the collision map are inflated by a certain radius. In the second step, Dijkstra's algorithm is employed to find the shortest, collision-free path.

In the next step, we thin out the resulting dense grid-sampled path. The dense path is approximated by linear straight-line segments. The approximation is based on comparing the path length to the segment length from the last added waypoint. If the difference exceeds a threshold, a new waypoint is added. By filtering the path in this way, implicitly higher velocities are reached in straight paths, while the velocity is reduced in areas with high curvature (e.g. around obstacles).

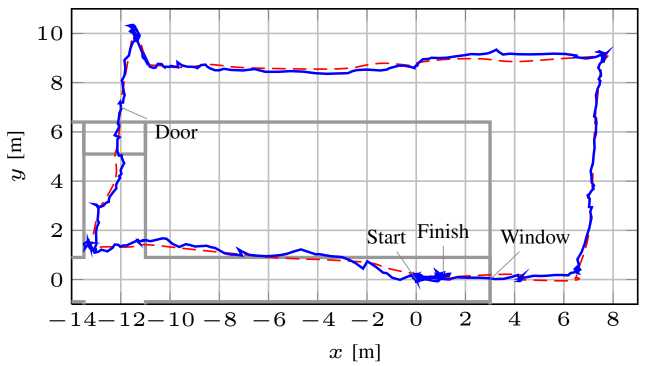
The path planner is configured to re-plan continuously, in order to react to previously unknown obstacles.

III. Experiments

For verifying the robustness of our system design, we chose a challenging indoor/outdoor flight path of about 60 m. We show the results of one flight representing the conducted 3 runs.

The transition from indoor to outdoor and vice versa is challenging in several aspects: the lighting conditions change quickly and usually the visual odometry shows dropouts for several images until the camera shutter is re-adapted. In contrast to our odometry system, feature based SLAM methods can easily lose the correspondence connection and need reinitialization. Wind conditions change suddenly between a narrow indoor corridor with self induced turbulence and a wide outside free space with possible wind gusts. In our case, we conducted the indoor/outdoor transition through a 1.26 m wide window, while the quadrotor has a diameter of 0.77 m. Therefore, the obstacle map has to be accurate to find a valid path through the inflated window frame and the controller has to follow the planned path precisely to prevent collisions.

During the experiment, the incrementally built on-board obstacle map was transmitted to a ground station where the operator clicked on the map to set a new goal point. As the starting point, we placed the quadrotor within the corridor



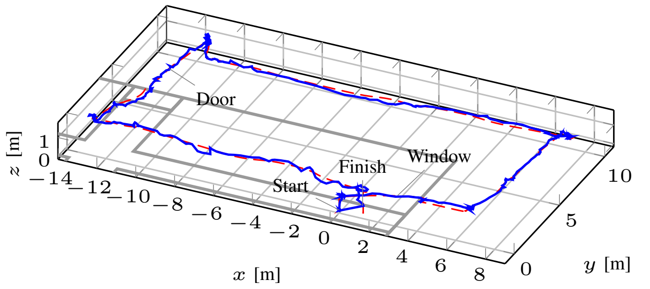


Fig. 7. Reference (- -) and estimated (—) path with locations of interesting points.

of a building. The operator commanded it to start and hover at a height of 1.6 m. The window became visible in the map and the operator set a waypoint outside the building. Figure 6 illustrates the on-board data processing chain just before crossing the window. Outside, the operator chose successive waypoints (partly in unknown map areas) to command the quadrotor around the building through the entrance back to the starting point. By continuously re-planning the flight path on the updated map, appearing obstacles were autonomously avoided by the quadrotor. If a waypoint set in unknown space was accidentally occupied by an obstacle, the system canceled the path in front of the obstacle and switched to hover mode waiting for a new goal.

A. Ego Motion Estimation

Figure 7 depicts the on-board estimated position in blue with the commanded reference trajectory in red. In the top view (top plot) the building construction plan is depicted in the background. The quadrotor flight was started in the middle of a 0.80×0.80 m platform defined as trajectory origin. At the end of the trajectory the quadrotor was commanded to land on the platform. The final, manually measured (x, y)-position was (0.15; 0.24) m, with a position estimate of (1.10; -0.23; -0.36) m (x,y,z). This corresponds to a total loop closure error of 1.13 m after a total trajectory length of about 60 m.

Sparse obstacle density on the trajectory (outside the building) is reflected in a sparse sampling of the commanded flight path. In these areas the quadrotor comes close to the maximum allowed flight speed of 1.8 m/s. Figure 8 shows the absolute quadrotor speed estimate in blue with the commanded reference trajectory in red.

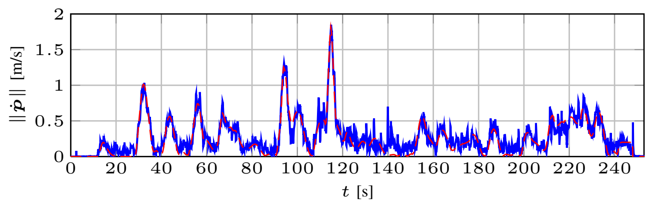


Fig. 8. Reference (- -) and estimated (—) flight velocity

B. Mapping

Figure 9 shows a 3D reconstruction of the flown area, recomputed offline at a higher resolution of 2 cm. The offline map was only processed for visualization in this paper. It is not used by the flying system itself. The depth images and ego-motion estimates which were calculated and logged on-board were used. No offline optimization or loop closure was applied. In Figure 10 we overlaid the reconstruction by the on-board calculated 3D obstacle layer at the altitude of the quadrotor in green. The inflated 2D obstacle map used for path planning is depicted in red.

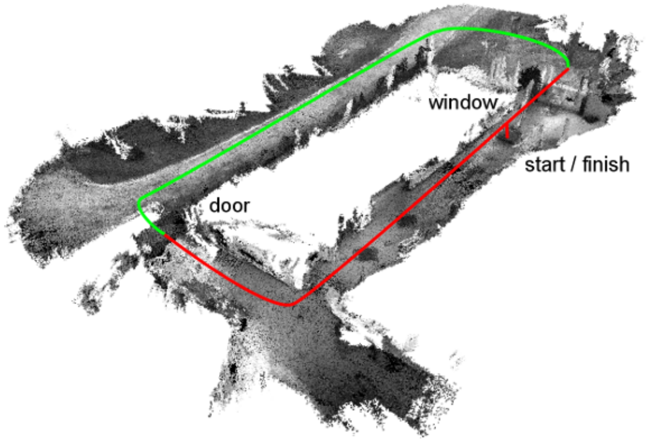


Fig. 9. Offline 3D reconstruction with 2 cm resolution using on-board calculated ego-motion estimates and depth data only. The indoor trajectory is marked in red, while the outdoor trajectory is marked in green.

IV. Discussion

The presented system is a major step in the direction of autonomous, flying robots that can be used as a surveillance tool in disaster management scenarios. All tasks including high resolution stereo image processing, visual odometry calculation, data fusion, mapping, path planning and control are realized in realtime on-board the robot. None of the algorithms requires special geometrical constraints on the surrounding environment such as flatness of the ground or vertical wall assumptions and the system works indoors as well as outdoors equally well. These are the basic requirements for real world SAR scenarios.

The conducted experiments show the robustness of our system in challenging situations as indoor/outdoor transitions with changing light and wind conditions. The accuracy

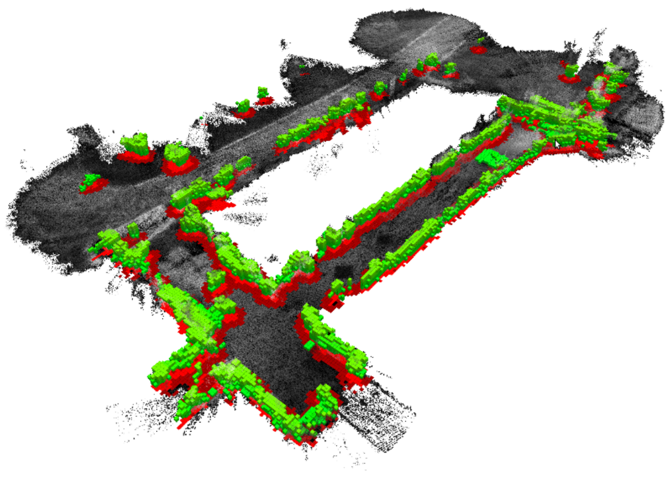


Fig. 10. Overlaid on-board 3D map view at 2 cm resolution. The obstacle layer at flying altitude and height of quadrotor is shown in green, while the down projected 2D inflated obstacle map used for planning is shown in red.

of the state estimation system with an experimental loop closure error of less than 2 % makes the system suitable for mixed indoor and outdoor exploration tasks. It has to be considered that the loop closure error is only used as accuracy measuremnt for our local navigation approach. We believe that global navigation in huge areas of operation can be realized more efficiently by combination of global topological navigation approaches [19] and our visual keyframe-based local, metric navigation in contrast to global pose graph SLAM with loop closure.

Nevertheless, one of the most critical factors for employing the system in real life scenarios is the limited flight time of less than 10 minutes due to limited battery capacity. Flight time can be increased by reducing the total system weight. We see great potential in lightening the navigation unit by combining all processing units on one base board. Considering the weight of the components we expect a possible reduction of at least 150 g. Furthermore, the system is limited to areas with good lighting conditions which could be solved by equipping the robot with active LED illumination.

At the current state, the system is on the limits of its computational load, therefore, we are optimizing our algorithms. We are working to move further parts of the vision pipeline into the FPGA. Candidates are image rectification, feature detection, description and initial correspondence determination. We assume a possible reduction of 50% of the CPU load caused by the vision pipeline. In the current implementation, our stereo odometry algorithm could be confused by moving objects covering great areas of the images. We plan to use data fusion results for supporting outlier detection in subsequent images.

Our system state estimation framework was shown in simulations to cope with fast flight dynamics doing maneuvers like flips [1]. We will explore the limits of our system dynamics in future work. Limitations will come from lighting conditions resulting in motion blur in the camera images and from physical motor thrust limitations. To guarantee safe, highly dynamic maneuvers in unknown environments we also have to integrate a fast, reactive collision avoidance as well as full 3D path planning.

V. Conclusion

We introduced a new, highly optimized quadrotor platform, developed for the use in disaster management and SAR scenarios. All navigation tasks including system state estimation, mapping and path planning are realized on-board the flying robot, which is the requirement for autonomy.

We developed a modular navigation unit which is integrated into our quadrotor as a monolithic, damped block with a stiffened configuration between cameras and IMU. The unit processes stereo images with a resolution of 0.5 MPixel using an FPGA implementation of the Semi-Global Matching algorithm at a rate of 14.6 Hz. Depth images are used for keyframe based stereo odometry calculation. The results are fused with IMU measurements, for compensating time delays of 250 ms that are introduced by the vision pipeline. System state estimation results are available with a rate of up to 819 Hz. They are employed for control and 3D mapping. The map forms the basis for dynamic 2D path planning for avoiding known and appearing obstacles.

We have demonstrated the autonomous navigation ability of our system by a challenging indoor/outdoor trajectory. Setting waypoints on the incrementally built on-board map, an operator commanded a roundtrip from inside a building corridor through the window around the building and through the door back to the starting point. Depending on the obstacle density, the system flies with a maximum speed of 1.8 m/s. The relative loop closure error of our metric, local navigation system on the 60 m trajectory was less than 2% without conducting an actual loop closure. The map built during the experiment, using on-board processed data only, is presented. The paper comes with a video of the flight.

In future work we will optimize system weight and implement further algorithms on the FPGA. In addition, we will increase system dynamics to demonstrate highly dynamic flight behavior.

A. *Acknowledgments*

We would like to thank Philipp Lutz (DLR) for his great system support during integration. Furthermore, we would like to thank Supercomputing Systems (SCS) in Switzerland as well as Uwe Franke and Stefan Gehrig (Daimler AG, Germany) for the FPGA implementation of SGM.

References

[1] K. Schmid, F. Ruess, M. Suppa, and D. Burschka, “State estimation for highly dynamic flying systems using key frame odometry with varying time delays,” in *IROS*, oct. 2012, pp. 2997 –3004.

[2] K. Schmid and H. Hirschmüller, “Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device,” in *IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[3] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, “Toward a Fully Autonomous UAV: Research Platform for Indoor and Outdoor Urban Search and Rescue,” *Robotics Automation Magazine, IEEE*, vol. 19, no. 3, pp. 46–56, 2012.

[4] H. Hirschmüller, "Stereo processing by semi-global matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, February 2008.

[5] A. Bachrach, A. De Winter, R. He, G. Hemnann, S. Prentice, and N. Roy, "RANGE Robust autonomous navigation in GPS-denied environments," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 1096–1097.

[6] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *2011 IEEE International Conference on Robotics and Automation*, vol. 10, no. March, GRASP Laboratory, University of Pennsylvania, Philadelphia, 19104, USA. IEEE, 2011, pp. 20–25.

[7] S. Shen, N. Michael, and V. Kumar, "Stochastic differential equation-based exploration algorithm for autonomous indoor 3D exploration with a micro-aerial vehicle," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1431–1444, Nov. 2012.

[8] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Fox, and N. Roy, "Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera," in *International Symposium on Robotics Research (ISRR)*, 2011, pp. 1–16.

[9] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2472–2477.

[10] F. Fraundorfer, L. Heng, D. Honegger, G. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-Based Autonomous Mapping and Exploration Using a Quadrotor MAV," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[11] D. Honegger, P. Greisen, L. Meier, P. Tanskanen, and M. Pollefeys, "Real-time velocity estimation based on optical flow and disparity matching," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, no. 1, 2012.

[12] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondance," in *Proceedings of the European Conference of Computer Vision*, Stockholm, Sweden, May 1994, pp. 151–158.

[13] H. Hirschmüller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1582–1599, September 2009.

[14] S. Gehrig, F. Eberli, and T. Meyer, "A real-time low-power stereo vision engine using semi-global matching," in *International Conference on Computer Vision Systems (ICVS)*, vol. LNCS 5815, Liege, Belgium, October 2009, pp. 134–143.

[15] H. Hirschmüller, P. R. Innocent, and J. M. Garibaldi, "Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics," in *Proceedings of the 7th International Conference on Control, Automation, Robotics and Vision*, Singapore, 2-5 December 2002, pp. 1099–1104.

[16] H. Hirschmüller, "Stereo vision based mapping and immediate virtual walkthroughs," Ph.D. dissertation, School of Computing, De Montfort University, Leicester, UK, June 2003.

[17] A. Stelzer, H. Hirschmüller, and M. Görner, "Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain," *International Journal of Robotics Research: Special Issue on Robot Vision*, vol. 31, no. 4, pp. 381–402, 2012.

[18] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013, software available at http://octomap.github.com. [Online]. Available: http://octomap.github.com

[19] M. Augustine, E. Mair, A. Stelzer, F. Ortmeier, D. Burschka, and M. Suppa, "Landmark-tree map: a biologically inspired topological map for long-distance robot navigation," in *Robotics and Biomimetics (ROBIO), 2012. IEEE International Conference on*. IEEE, 2012.