Vision-Controlled Micro Flying Robots

From System Design to Autonomous Navigation and Mapping in GPS-Denied Environments

By Davide Scaramuzza, Michael C. Achtelik, Lefteris Doitsidis, Friedrich Fraundorfer, Elias Kosmatopoulos, Agostino Martinelli, Markus W. Achtelik, Margarita Chli, Savvas Chatzichristofis, Laurent Kneip, Daniel Gurdan, Lionel Heng, Gim Hee Lee, Simon Lynen, Lorenz Meier, Marc Pollefeys, Alessandro Renzaglia, Roland Siegwart, Jan Carsten Stumpf, Petri Tanskanen, Chiara Troiani, and Stephan Weiss

Autonomous microhelicopters will soon play a major role in tasks like search and rescue, environment monitoring, security surveillance, and inspection. If they are further realized in small scale, they can also be used in narrow outdoor and indoor environments and represent only a limited risk for people. However, for such operations, navigating based only on global positioning system (GPS) information is not sufficient. Fully autonomous operation in cities or other dense environments requires microhelicopters to fly at low altitudes, where GPS signals are often shadowed, or indoors and to actively explore unknown environments while avoiding collisions and creating maps. This involves a number of challenges on all levels of helicopter design, perception, actuation, control, and navigation, which still have to be solved. The Swarm of Micro Flying Robots (SFLY) project was a European Union–funded project with the goal of creating a swarm of vision-controlled microaerial vehicles (MAVs) capable of autonomous navigation, three-dimensional (3-D) mapping, and optimal

Digital Object Identifier 10.1109/MRA.2014.2322295
Date of publication: 20 August 2014

1070-9932/14©2014IEEE

surveillance coverage in GPS-denied environments. The SFLY MAVs do not rely on remote control, radio beacons, or motion-capture systems but can fly all by themselves using only a single onboard camera and an inertial measurement unit (IMU). This article describes the technical challenges that have been faced and the results achieved from hardware design and embedded programming to vision-based navigation and mapping, with an overview of how all the modules work and how they have been integrated into the final system. Code, data sets, and videos are publicly available to the robotics community. Experimental results demonstrating three MAVs navigating autonomously in an unknown GPS-denied environment and performing 3-D mapping and optimal surveillance coverage are presented.

Figure 1. The three SFLY hexacopters are designed for inertial–visual navigation in GPS-denied environments.

Motivation

Autonomous navigation of microhelicopters (where *micro* means up to the size of a few decimeters and fewer than 2 kg) has progressed significantly in the last decade thanks to the miniaturization of exteroceptive sensors (e.g., laser rangefinders and digital cameras) and to the recent advances in microelectromechanical systems, power supply, and vehicle design.

Microhelicopters—and, notably, multirotor helicopters—have several advantages compared with fixed-wing microaerial vehicles: they are able to take off and land vertically, hover on a spot, and even dock to a surface. This capability allows them to easily work in small indoor environments, pass through windows [1], traverse narrow corridors, and even grasp small objects [2].

A key problem in aerial-vehicle navigation is the stabilization and control in six degrees of freedom (DoF), i.e., attitude and position control. Today's systems handle the attitude control well. However, without a position control, they are prone to drift over time. In GPS-denied environments, this can be solved using offboard sensors (such as motion-capture systems or total stations) or onboard sensors (such as cameras and laser rangefinders). The use of offboard sensors allows research to focus on control issues without dealing with the challenges of onboard perception. Today's popular MAV testbeds are made by Vicon or OptiTrack motion-capture systems, which consist of multiple infrared static cameras tracking the position of a few highly reflective markers attached to the vehicle with millimeter accuracy and at a very high frame rate (several hundred hertz). These systems are very appropriate for testing and evaluation purposes [3], such as multirobot control strategies or fast maneuvers, and serve as a ground-truth reference for other localization approaches. Using this infrastructure, several groups have demonstrated aggressive maneuvers and impressive acrobatics [4], [5]. In the works mentioned previously, the MAVs are actually blind. To navigate, they rely on the highly precise position measurement provided by the external motion-tracking system. As a matter of fact, what is really autonomous is not the single MAV itself but the system comprising the MAVs plus the external cameras. Furthermore, these systems are limited to small, confined spaces, and require manual installation and calibration of the cameras, making it impossible to navigate autonomously in unknown, yet-unexplored environments. Therefore, for a MAV to be fully autonomous, sensors should be installed on board.

Contributions of SFLY

This article describes the technical challenges and results of the three-year European project SFLY (www.sfly.org) devoted to the implementation of a system of multiple microflying robots capable of autonomous navigation, 3-D mapping, and optimal coverage in GPS-denied environments. The SFLY MAVs can fly using only an onboard camera and an IMU. This article describes the major contributions of the SFLY, from hardware design and embedded programming to vision-based navigation and mapping. The first contribution is the development of a new hexacopter equipped with enough processing power for on board computer vision (Figure 1). The second contribution is the development of a local-navigation module based on monocular simultaneous localization and mapping (SLAM) that runs in real time on board the MAV. The output of the monocular SLAM is fused with inertial measurements and is used to stabilize and control the MAV locally without any link to a ground station. The third contribution is an offline dense-mapping process that merges the individual maps of each MAV into a single, global map that serves as input to the global navigation module. Finally, the fourth contribution is a cognitive, adaptive optimization algorithm to compute the positions of the MAVs, which allows the optimal surveillance coverage of the explored area.

Related Work

System Design

Extensive work has been carried out on quadrotor systems. The function principle of quadrotors can be found in [6] and [7]. A review of the state of the art on modeling, perception, and control of quadrotors can be found in [8]. The pitch angle of the propellers is typically fixed; an evaluation of variable-pitch propellers is presented in [9]. The platform described in this article, the AscTec FireFly, is an improvement of the previous and popular model known as AscTec Pelican. While

other groups often run the computation off board, by transmitting image data to a powerful ground-station computer, the SFLY platform runs most computer-vision algorithms fully on board. This demands high onboard-computation capabilities. In the first SFLY vehicle [10], a 1.6-GHz Intel Atom computer was used; however, in the latest platform, this was replaced with a Core 2 Duo onboard computer able to process all flight-critical data on board.

Autonomus Navigation

Autonomous navigation based on onboard two-dimensional (2-D) laser rangefinders has been largely explored for ground mobile robots [11]. Similar strategies have been extended to MAVs to cope with their inability to "see" outside the scan plane. This is usually done by varying the height and/or the pitch and roll of the helicopter as well as by incorporating readings from air-pressure and gyroscopic sensors [1], [12]–[16]. Although laser scanners are very reliable and robust, they are still too heavy and consume too much power for lightweight MAVs. Therefore, vision sensors are very appealing; however, they require external illumination and a certain computing power to extract meaningful information for navigation.

To avoid drift over long periods of time, the system should be able to relocalize whenever it comes back to a previously visited location.

Most of the research on vision-based control of MAVs has focused on optical flow [17]–[19]. But since optical flow can only measure the relative velocity of features, the position estimate of the MAV will inevitably drift over time. To avoid drift over long periods of time, the system should be able to relocalize whenever it comes back to a previously visited location. One possibility is offered by SLAM approaches.

Preliminary experiments for MAV localization using a visual extended Kalman filter (EKF)-based SLAM technique were described in [20]. However, the first use of visual SLAM to enable autonomous basic maneuvers, such as takeoff and landing, point-to-point navigation, and drift-free hovering on the spot, was done right within the framework of the SFLY project [21], [22]. Due to the use of a single camera, the absolute scale was initially determined manually or using a known-size object [23]. Later, the system was extended [24] to incorporate data from an IMU and, thus, estimate the absolute scale automatically while self-calibrating all the sensors (this approach will be outlined in the "Inertial-Aided Visual Navigation" section).

Optimal Coverage

Optimal coverage is the problem of computing the poses of a team of robots, which guarantee the optimal visibility of an area under the constraints that

- the part of terrain monitored by each robot is maximized [25], [26]
- for every point in the terrain, the closest robot is as close as possible to that point.

The second objective is necessary for two practical reasons: 1) the closer the robot is to a point in the terrain, the better its sensing ability to monitor this point, and 2) in many multirobot coverage applications, there is the necessity of being able to intervene as fast as possible in any of the points of the terrain with at least one robot. The optimal visibility problem is also related to the art-gallery problem, where the goal is to find the optimum number of guards in a nonconvex environment such that each point of the environment is visible by at least one guard [27], [28]. An incremental algorithm, which also considers a maximum monitoring distance, was presented in [29], while the optimal coverage of a 2-D region with a team of flying robots was studied in [30].

Most approaches for multirobot surveillance coverage concentrate on the second objective and tackle 2-D surfaces [31]–[35]. A method for nonplanar surfaces embedded in 3-D was presented in [36], while a study for multiple flying robots equipped with downward-looking cameras observing a planar 2-D environment was proposed in [30]. Multirobot optimal-coverage algorithms for convex environments were proposed in [31] and [32] using Voronoi partition, while in [33] the classical Voronoi coverage was combined with the Lloyd algorithm and the local path-planning TangentBug algorithm. In [32], a function indicating the relative importance of different areas in the environment using information from onboard sensors was used. An approach for nonconvex environments was proposed in [34] using Voronoi partition and in [35] using the potential-field method. Partition was obtained using the geodesic distance instead of the Euclidean one, considering the particular topology of the problem. In [35], the same problem was approached using the potential-field method. Another possible solution for convex environments with obstacles was proposed in [33]: the classical Voronoi coverage was combined with the Lloyd algorithm and the local path-planning TangentBug algorithm.

In all of the aforementioned approaches, the regions to monitor are considered in two dimensions. An approach for 3-D spaces was proposed in [36]. Conversely, the approach described in this paper is based on a new stochastic optimization method, called *cognitive-based adaptive optimization* (*CAO*). This method addresses 3-D environments and tackles the two aforementioned objectives simultaneously.

Microhelicopter Platform

Design Concept

One goal of the SFLY project was to have a vehicle as small, lightweight (fewer than 1.5 kg), and safe as possible, while being capable of carrying and powering an onboard computer and cameras. Since the SFLY helicopter was envisaged to operate in urban environments, the impact energy had to be reduced to a minimum. To limit the risk of injuries, studies

were made to evaluate the effects of having more than four (but smaller and safer) rotors on efficiency achievable dynamics and redundancy. These studies are presented in detail in [37]. In brief, it was found that the smaller the number of rotors, the better the efficiency of the vehicle. On the other hand, the achievable dynamics and, therefore, the maneuverability of the vehicle can be enhanced by a larger number of propellers and a smaller ratio between rotor surface and total weight. However, for safe operation, the most important aspect is redundancy against at least a single-rotor failure. In [37], it was shown that the minimum number of rotors with redundancy against a single failure could be reduced to six due to a new redundancy concept. To do so, different shapes of redundant multirotor vehicles were analyzed, and the maximum thrust in a redundancy situation was calculated. The results are shown in Table 1 (neglecting the additional margin needed to control the other axes). The hexagon-shaped six-rotor design was chosen as the best tradeoff. By deriving a control scheme for such a configuration, it can be concluded that, if all working motors are to spin at least at idle speed, a six rotor helicopter in hexagon shape cannot compensate for a single-motor failure. Undesired momentum around the yaw axis will be the result if pitch and roll are to be controlled (see [37] for more details). To overcome this disadvantage, a new and very simple control scheme was developed, which is shown in Figure 2.

The selected configuration can be built with propellers as small as the known safe propellers of the AscTec Hummingbird [10]. In addition, it can carry the demanded payload and is redundant against single-rotor failures, thus enabling safe operations in urban areas. Compared with an octocopter design, the thrust in a redundancy situation is smaller but the overall efficiency is higher due to the use of six rotors instead of eight.

Table 1. The theoretical maximum thrust in redundancy situations for different configurations.

System Configuration	Thrust in Failure Situation
Triangle hex	50%
Hexagon hex	66%
V-shape octo	62%
Octagon octo	70–73%

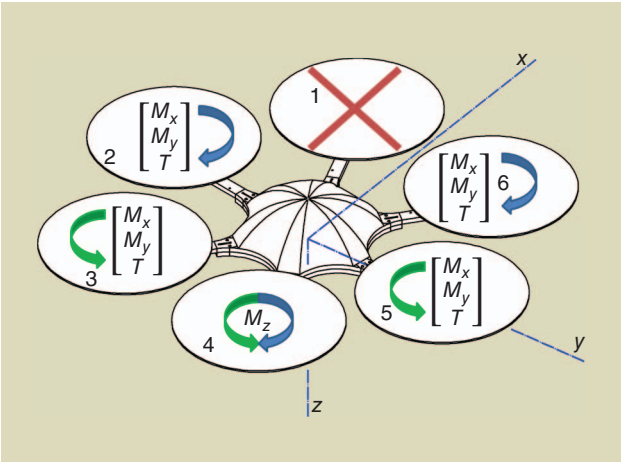

Figure 2. This illustration shows the redundancy against single-rotor failure. Assuming that motor 1 is failing, motors 2, 3, 5, and 6 are controlled by the thrust command and the roll and pitch controllers' output. Motor 4—on the opposite side of the failing motor—compensates and controls the yaw momentum by repeatedly changing its direction.

Electronic Architecture

Except for the two additional motors, the electronic components and the software architecture are about the same as the AscTec Pelican described in [10].

To handle motor-failure situations, the communication protocols were extended for six motors, and the algorithms for failure detection and redundancy handling added. The control scheme for these failure situations is prescripted for each one of the possible six failure cases to be activated automatically if a failure is detected.

A distribution of the flight-control units (FCUs) main task between two microprocessors is shown in Figure 3. The so-called

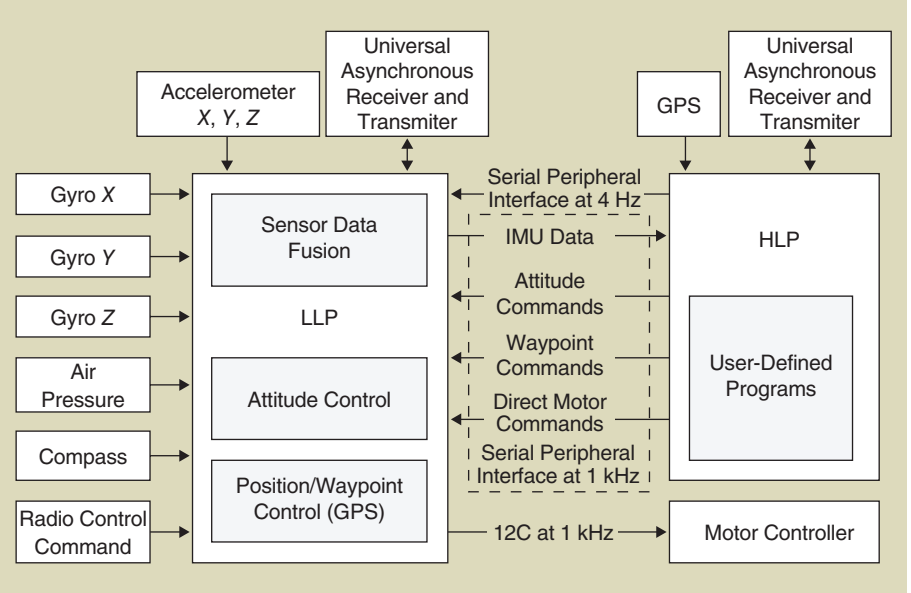

Figure 3. The electronic architecture: all sensors, except the GPS, are connected to the LLP, which communicates via I2C with the motor controllers and via the serial peripheral interface with the HLP. For this article, position control and state estimation for local navigation are implemented in the "user-defined programs" section of the HLP (see Figure 8). The position (waypoint) controller on the LLP is not used here since we work in GPS-denied environments.

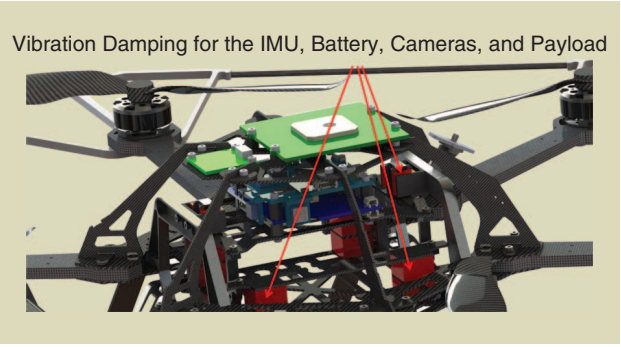

Figure 4. The computer-aided design (CAD) model illustrating the vibration damping between the two parts of the frame: the motors and the landing gear are connected to the outer frame, and the inner frame to the IMU, battery, and payload. The silicon dampers are highlighted in red.

low-level processor (LLP) handles all hardware interfaces; it is connected to the sensors and computes the attitude-data-fusion and flight-control algorithms at an update rate of 1 kHz. The high-level processor (HLP) is open for customized or experimental code. In the SFLY project, the HLP is used for state estimation and control. It has proven to be helpful to have the LLP as a safety backup while performing experiments in flight.

The SFLY MAVs do not rely on remote control, radio beacons, or motion-capture systems.

Onboard Computer

To integrate all computationally intense parts on board the vehicle, the initial Atom computer board of the Pelican platform was not sufficient. Therefore, the ongoing development of a new motherboard supporting the COM Express standard was pushed forward to support the integration of a Dual Core Atom, a Core 2 Duo, or a Core i7 central processing unit. These computer boards provide enough computational power to run all onboard software. Furthermore, additional interfaces like Firewire and hardware serial ports are supported. Specifically, the hardware serial ports are another step toward precise and fast state estimation on the onboard computer, as the latency is reduced to a minimum.

Mechanical Concept and Vibration Decoupling

One important requirement, raised from test flights of the previous vehicles, is a vibration decoupling. Just decoupling the IMU has proven to be insufficient. Instead, payloads such as cameras should be decoupled as well and, ideally, fixed to the IMU. Vibration damping is necessary to improve state estimation for position control as well as image quality. The damping system has to be designed so that there is a rigid connection between the cameras and IMU to avoid any dynamic misalignment. These requirements led us to a completely new concept. A so-called frame-in-frame concept was built: the outer frame holds the motors, the landing gear, the canopy, and the propeller protection, while the inner frame carries the IMU, the battery, and the payload. As shown in Figure 4, both frames are connected using special silicon dampers, distributed in a pattern to intentionally influence the dynamics between both frames. This is necessary because the frame-in-frame concept leads to additional dynamics between both parts. The eigenmodes of this new dynamic system had to be adjusted so that no resonance oscillations between both frames occurred for a variety of payload configurations. Flight tests show an improvement of image and state-estimation quality, and all resonance oscillations are eliminated. Due to this new damping concept, the whole mechanical structure had to be redesigned.

To reduce the overall height and to concentrate the mass closer to the center of gravity, the battery was moved to the center of the frame. Furthermore, a landing gear was added to protect the payload, which is connected to the dampened frame. A rollover bar protecting the electronic components and supporting the cover was added as well.

In addition to these additional features, another requirement was to enable fast component changes in case of a crash or modification during integration and testing. To put all these requirements and features together, a new combination of carbon fiber, carbon fiber sandwich, and aluminum was chosen.

Details of this concept can also be observed in Figure 4, and a complete computer-aided design (CAD) model, including a camera mount, is shown in Figure 5. Note that only one camera (downward looking) is used for navigation, while the other two, in stereo configuration, are used for obstacle avoidance (not described here) and dense matching (see the "3-D Mapping" section). Table 2 summarizes the main technical data.

Figure 5. The complete CAD model, including three cameras on the SFLY hexacopter.

Table 2. The main technical data.

Empty weight without battery	640 g
$I_{xx} \approx I_{yy}$	0.013 kg·m^2
I_{zz}	0.021 kg·m^2
Total thrust (at 10.5 V)	24 N
Maximum takeoff weight	1,090 g
Maximum payload	450 g
Maximum flight time	Up to 30 min

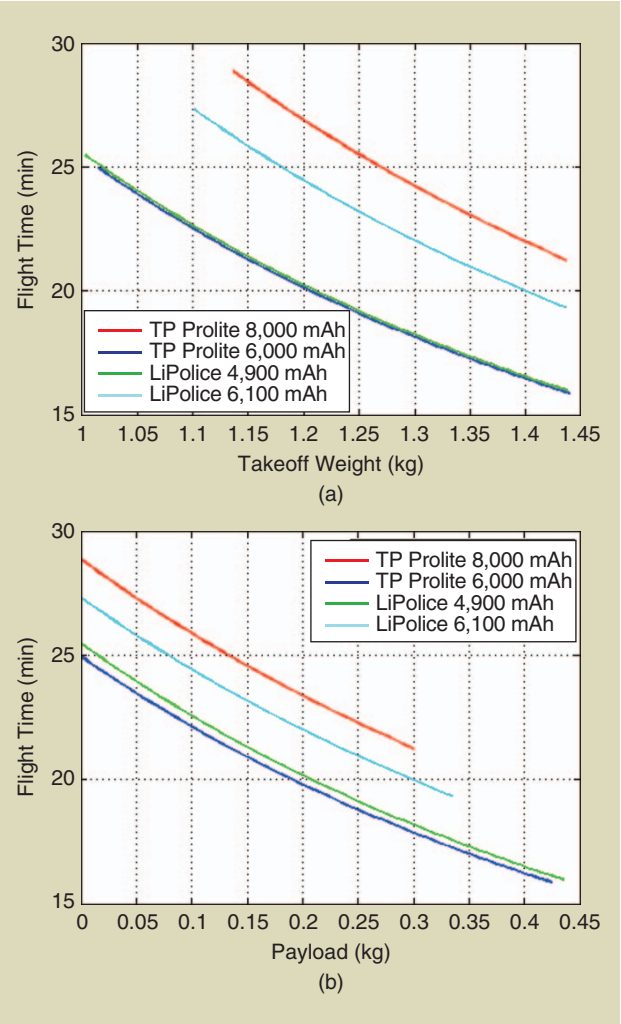

Figure 6. The calculated flight time is plotted versus (a) takeoff weight and (b) payload. The estimated flight time for a given payload with different batteries is shown. Thunder Power (TP) and LiPolice are LiPo battery manufacturers.

Flight-Time Estimation and Payloads

Based on test-bench data of the consequently improved motors and propellers as well as a final empty weight of 640 g, the flight time can be calculated for different payloads and batteries (Figure 6). The weight of the different batteries is considered, and the plots are limited to the maximum takeoff weight. The flight time is calculated for 85% of the battery capacity because lithium-polymer batteries must not be completely discharged. For the SFLY requirements, a 4,900-mAh battery was selected, resulting in an approximately 16-min flight time at a 400-g payload (neglecting the onboard-computer power consumption).

Inertial-Aided Visual Navigation

The navigation of the MAVs is handled by two different modules that are named *local navigation* and *global navigation.* The local-navigation module is responsible for flight stabilization, state estimation (including absolute-scale estimation), and waypoint-based navigation of each MAV. It runs on board each platform and estimates the pose of each MAV with respect to its starting position and, hence, does not rely on a persistent connection to the ground station. The state estimator and position controller are spread over the different computation platforms, according to the processing power and to reduce delay. The relative positions of the MAVs at start are unknown. The task of the global-navigation module (running off board the MAVs, on a ground-station computer) is to express the poses of all MAVs in a common, global coordinate frame and, possibly, to reduce both motion and map drifts. This is done by identifying both loop closures by the same MAV and path intersections between multiple MAVs (Figure 7). The interaction between these modules can be observed in Figure 8.

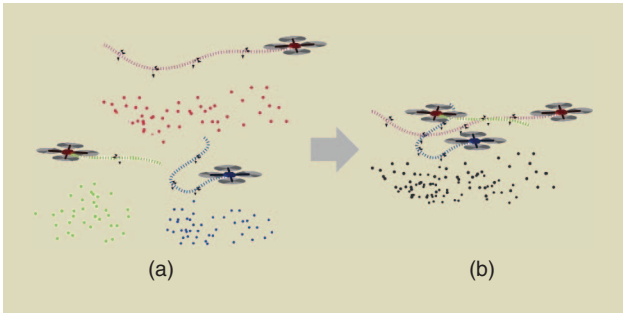

Figure 7. (a) The local-navigation module (running on board) estimates the pose of each MAV independently for each platform. (b) The global-navigation module (offboard) recognizes path intersections and uses them to express the MAVs' poses in the same, global coordinate frame and to reduce drift.

Local Navigation

In recent years, 5-DoF single-camera-based visual odometry (VO) has made significant progress. (a tutorial on monocular and stereo VO can be found in [38] and [39]). (Here, we refer to 5 DoF instead of 6 DoF because the absolute scale is not observable with a single camera. However, the scale factor can be estimated by adding an IMU, as explained in this section.) Filter-based and key-frame-based off-the-shelf algorithms are publicly available. Because of its robustness, real-time performance, and position accuracy, the key-frame-based solution proposed in [40] was selected and tailored to the general needs of our computationally limited platform. However, nowadays, more recent VO algorithms, such as Semidirect Visual Odometry [41], represent a more robust, more accurate, and faster option for MAVs.

The local-navigation module is responsible for flight stabilization, state estimation, and waypoint-based navigation of each MAV.

Our framework uses the Robot Operating System (ROS) middleware (www.ros.org) and runs on a standard Ubuntu operating system, facilitating the development of

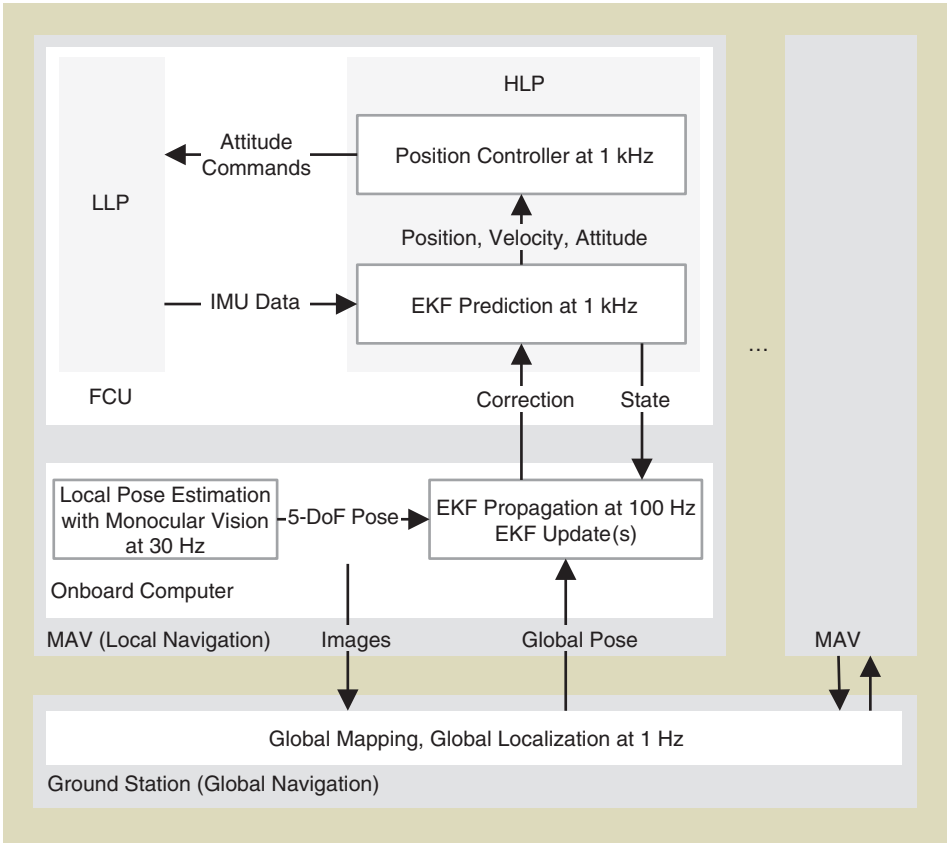

Figure 8. An overview of the different processing tasks and how these are distributed and interact with each other in our navigation framework. Waypoint commands are sent from the ground station to the onboard computer, which forward them to the position controller. The ground station and onboard computer communicate over a Wi-Fi connection. Note that all critical parts necessary to keep the helicopter airborne run entirely on board and do not rely on the Wi-Fi link. The parts on the HLP refer to the "user-defined programs" section in Figure 3.

new algorithms. The current implementation uses only 60% of one core of the Core 2 Duo processor at 30 Hz, leaving enough resources for future higher-level tasks. As a reference, the same implementation on an Atom 1.6-GHz single-core computer runs at 20 Hz using 100% of the processing power.

The 5-DoF pose of the MAV camera output by the visual-odometry algorithm was fused with the inertial measurements of an IMU using an EKF. More details are given in [42]. An EKF framework consists of a prediction and an update step. The computational load required by these two steps is distributed among the different units of the MAV, as described in [43]. The state of the filter is composed of the position p^i_w, the attitude quaternion q^i_w, and the velocity v^i_w of the IMU in the world frame. The gyroscope and accelerometer biases b_ω and b_a as well as the missing scale factor λ are also included in the state vector. For completeness, the extrinsic calibration parameters describing the relative rotation q^s_i and position p^s_i between the IMU and the camera frames were also added. Note that the calibration parameters could be omitted from the state vector and be set to a premeasured constant to increase robustness and faster state convergence. Having p^s_i as a constant and not as a filter state would increase the convergence performance of the visual scale. We did not notice significant performance improvement when removing q^s_i from the state vector. We assume that this is because the attitude is directly measured by the visual pipeline whereas the scale ambiguity occurs in both p^s_i and the MAV position p^i_w. In this article, we show that even with continuously estimating both parameters p^s_i and q^s_i, we can achieve good and robust results. This yields a 24-element state vector X:

$$X = \{p^{i^T}_w \; v^{i^T}_w \; q^{i^T}_w \; b^T_\omega \; b^T_a \; \lambda \; p^{s^T}_i \; q^{s^T}_i\}. \tag{1}$$

Figure 9 shows the setup with the IMU and camera coordinate frames and the state variables introduced above.

The equations of the EKF prediction step for the considered IMU-camera fusion are given in [42]. The equations of the update step are derived by computing the transformation from the world reference frame to the camera frame as follows. For the position z_p, we can write:

$$z_p = p^s_w = (p^i_w + C^T_{(q^i_w)} p^s_i)\,\lambda + n_p, \tag{2}$$

where $C_{(q^i_w)} \in SO(3)$ is the rotation matrix associated with the IMU attitude quaternion q^i_w in the world frame, z_p denotes the observed position (the output of the visual odometry), λ is the scale factor, and n_p the measurement noise. For the rotation measurement z_q, we apply the notion of error quaternion. Since the visual-odometry algorithm yields the rotation q^s_w from the world frame to the camera frame, we can write:

$$z_q = q^s_w = q^s_i \otimes q^i_w. \tag{3}$$

For safe operation, the most important aspect is redundancy against at least a single-rotor failure.

A nonlinear observability analysis [44] reveals that all state variables are observable, including the intersensor calibration parameters p_i^s and q_i^s. Note that the visual pose estimates are prone to drift in position, attitude, and scale with respect to the world-fixed reference frame. Since these quantities are observable (and, notably, roll, pitch, and scale), gravity-aligned metric navigation becomes possible even in long-term missions. This is true as long as the robot excites the IMU accelerometer and gyroscopes sufficiently, as discussed in [45]. Note that the estimated attitude and position of the MAV in the world frame is subject to drift over time. However, since the gravity vector measured by the IMU is always vertically aligned during hovering, this prevents the MAV from crashing due to gravity misalignment even during long-term operations.

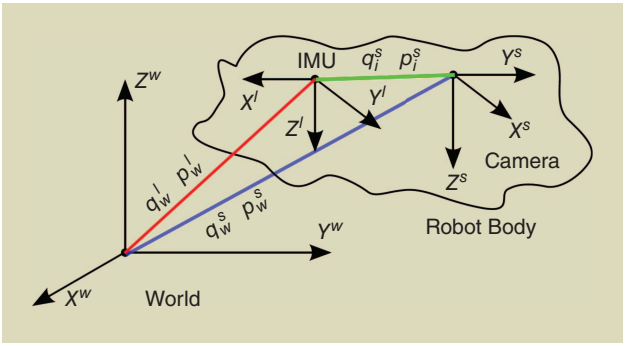

Figure 9. A setup depicting the robot body with its sensors with respect to a world reference frame. The system state vector is $X = \{p_w^i\, v_w^i\, q_w^i\, b_\omega\, b_a\, \lambda\, p_i^s\, q_i^s\}$, whereas p_w^s and q_w^s denote the robot's sensor measurements in (a possibly scaled) position and attitude, respectively, in a world frame.

Global Navigation

The task of the global-navigation module (running on the ground station) is to express the poses of all MAVs in a common, global coordinate frame and, possibly, to reduce both motion and map drifts. This is done by matching the current camera image to a 3-D environment map. The 3-D map consists of landmarks (3-D points and corresponding descriptors in each image) and the corresponding camera poses. The 3-D map is computed offline, as described in the "3-D Mapping" section and combines the maps of the individual MAVs into a single merged map. Map merging works by identifying both loop closures by the same MAV and path intersections between multiple MAVs (Figure 7). To reduce the computational load of the onboard computer, the global-navigation module runs on a ground station that constantly receives the images of the MAVs via Wi-Fi and sends back the updated global poses. To save bandwidth, a valid alternative to sending full camera frames is to have each MAV stream only features of selected key frames and relative-pose estimates, as recently proposed in [46].

Matching the current camera view to the 3-D map is done by vocabulary-tree-based image search, as described in [47]and [48]. For every frame, speeded-up robust Features [49] are extracted and then quantized into visual words using a vocabulary tree that was pretrained on a general image data set. The image IDs and the corresponding visual words are stored in a database that is organized as an inverted file for efficient data access. Additional metadata (pose estimates from the local-navigation module and IMU data) are stored with each image in the database. Whenever a new image is processed, it is ranked with all images in the database according to the similarity of the visual words. Geometric verification is performed on the top-N most similar frames using perspective three-point algorithm-based random sample consensus (RANSAC) [50]. A match is accepted if the inlier count exceeds a certain threshold. The initial pose from RANSAC is refined using nonlinear optimization and is sent back as global pose update. This approach allows for efficient localization and also scales to large maps.

3-D Mapping

For the 3-D mapping of the environment, an offboard ground station takes images from all MAVs and fuses them offline into a detailed map. The mapper is based on the general framework for graph optimization (g2o) framework [51]; it uses a pose-graph optimizer for prealignment of the data and then runs a bundle adjustment to get optimal results.

The maximum-likelihood estimates of the poses are computed by minimizing the Euclidean distances between the transformations in a pose graph. The nonlinear optimization is done by sparse Cholesky decomposition using the g2o framework. To improve the accuracy of the map, a bundle adjustment is run. The bundle adjustment optimizes the poses and the 3-D positions of all features at the same time by minimizing the image reprojection error. The corresponding graph of this problem consists of the MAV poses and the 3-D feature points as nodes. They are connected by edges that represent the projection of the 3-D feature point to images where the feature was detected. During the loop-detection phase, for every new frame, all image projections of the inlier features are added to the bundle-adjustment graph.

The outer frame holds the motors, the landing gear, the canopy, and the propeller protection.

A dense map is built offline using the poses of the MAV computed from the bundle adjustment process and the corresponding stereo images. For each pose and corresponding stereo frame, a 3-D point cloud in global coordinates is computed via stereo triangulation and used to update a 3-D occupancy map. After all the data have been processed, a terrain map is extracted from the 3-D occupancy map by thresholding the occupancy value in each cell in the occupancy map. The terrain map is triangulated to create a dense mesh. Furthermore, a dense textured map is created by projecting all triangular faces in the mesh onto the images, from which the faces are entirely visible, and texturing each face with the image that has the smallest

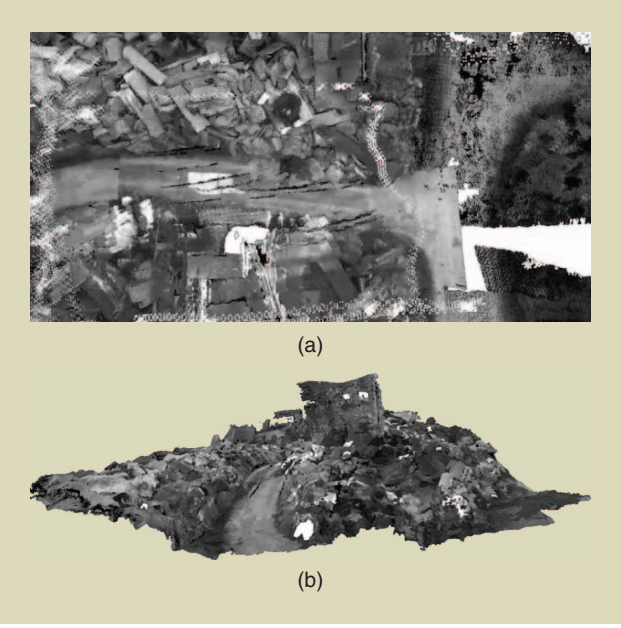

Figure 10. The textured visualization of the 3-D map of the firefighter area: (a) top and (b) side views.

incident angle relative to the face normal. This image-selection heuristic helps to minimize perspective distortion. A textured visualization of a 3-D map is shown in Figure 10. More details can also be found in [52].

Optimal Surveillance Coverage

The problem of deploying a team of flying robots to perform surveillance coverage missions over an unknown terrain of complex and nonconvex morphology was tackled using a novel CAO algorithm. The CAO algorithm was originally developed and analyzed for the optimization of functions for which an explicit form is unknown but measurements are available as well as for the adaptive fine-tuning of large-scale nonlinear-control systems [53]. The many advantages of using stochastic gradient descent algorithms, like the simultaneous perturbation stochastic approximation algorithm [54], to approach a sensor-based deployment problem have already been highlighted in [55].

The mapper is based on the general framework for graph optimization framework.

Within SFLY, CAO was implemented for surveillance tasks in unknown 3-D terrains of complex and nonconvex morphology with obstacles using only onboard monocular vision. CAO possesses several advantages compared with the previous works described in the "Optimal Coverage" section: it is computationally simple to implement, scalable, and can easily embed any kind of physical constraints and limitations (e.g., obstacle avoidance, nonlinear sensor-noise models). CAO does not create an approximation or estimation of the obstacles' location and geometry; conversely, it produces an online local approximation of the cost function to be optimized. A detailed description of the CAO algorithm and its functionality for the case of a team of aerial robots can be found in [56] and [57].

In the context of the SFLY project, the CAO algorithm tackles two objectives simultaneously to assure that the robot team will perform optimal surveillance coverage:

- maximize the part of terrain monitored by each robot
- for every point in the terrain, the closest robot has to be as close as possible to that point.

If only the first objective were considered, the robots would fly as high as their visibility threshold allows (which is defined as the maximum distance the robot's sensor can measure). Therefore, the second objective ensures that, among all possible configurations that maximize the visible area V, the robot team converges to the one that keeps as small as possible the average distance between each robot and the part of the terrain for which that particular robot is responsible. (Note that in the ideal case, where there are no limits for the robot's maximum height, and the robot has unlimited sensing capabilities, it suffices to have a single robot at a very high position to monitor the whole terrain.) The second objective is also necessary for two practical reasons: first, the closer the robot is to a point in the terrain, the better, in general, its sensing ability to monitor this point; second, in many multirobot coverage applications, it is necessary to intervene as fast as possible in any of the points of the terrain with at least one robot.

The two aforementioned objectives are combined in an objective function that the robot team has to minimize [56], i.e.,

$$J(P) = \int_{q \in V} \min_{i \in \{1,\ldots,M\}} \| x^{(i)} - q \|^2 dq + K \int_{q \in T-V} dq, \qquad (4)$$

where M is the number of robots that are deployed to monitor a terrain T, $x^{(i)}$ is the position of the ith robot, $P = \{x^{(i)}\}_{i=1}^{M}$ denotes the configuration of the robot team, q is a point in the terrain T, V consists of all points $q \in T$ that are visible from the robots, and K is a user-defined positive constant.

The first term in (4) addresses the second objective. The second term addresses the first objective and relates to the invisible area of the terrain (i.e., $\int_{q \in T-V}$, which is the total part of the terrain that is not visible to any of the robots). The positive constant K serves as a weight to give more or less priority to one or the other objective. A detailed analysis of the effect of K is presented in [56].

The implementation of CAO within the SFLY framework ensures that the physical constraints are also met throughout the entire multirobot coverage application. Such physical constraints include, but are not limited to, the following:

- the robots remain within the terrain's limits
- the robots satisfy a maximum-height requirement while not hitting the terrain
- the robots do not come closer to each other than a minimum allowable safety distance.

The above constraints can be easily formulated and incorporated in the optimization problem [56]. CAO uses function approximators for the estimation of the objective function at each time instant; therefore, a crucial factor for the successful implementation is the choice of the regressor vector, as described in [56]. Once the regressor vector has been set and the values of the cost function are available for measurement, it is possible to find at each time step the vector of parameter estimates and, thus, the approximation of the cost function.

Experimental Results

Flying Platform

The achievable dynamics and maneuverability are demonstrated by the accurate trajectory following and position control shown in Figure 11. To evaluate the redundancy capabilities, a switch disabling one motor was implemented to simulate a motor failure. There was no measurable deviation in the roll and pitch axes, but the maximum thrust is obviously limited during this failure situation.

Figure 12 shows the motor commands input to the four propellers during such a redundancy test. The motor commands are in the range [–100, 200]. As observed, at about 14 s, one motor is deactivated (the yellow plot drops to zero), and one motor command starts compensating for the yaw moment by slowly oscillating around zero (red plot). The other four motors are set feedforward to a higher thrust to compensate for the loss caused by the other two motors. Figure 12(b) shows the pilot's stick inputs. This plot looks absolutely normal for a manual flight, like Figure 12(c), which shows the attitude measure.

Vision-Based Navigation

Figures 13 and 14 show the evolution of the position and attitude of one MAV estimated by the EKF framework described in the "Local Navigation" section. The position plot (Figure 13) shows that the visual scale has been estimated correctly by the filter throughout the whole flight; as can be observed, the position and attitude drifts of the vision system are very low. For a rapidly drifting vision system, one would observe an increased difference between the GPS data and filter estimates. Note that GPS measurements were used during the initialization phase to align all states for a simpler comparison with ground truth. After this alignment phase (at about $t = 80$ s in Figures 13 and 14), GPS was no longer used as additional input in the EKF framework.

A 350-m trajectory estimated using this framework, resulting in an overall position drift of only 1.5 m, is shown in Figure 15. The presented framework was tested under a variety of challenging conditions, exhibiting robustness in the presence of wind gusts, strong light conditions causing saturated images, and large-scale changes in flight altitude. More details are given in [58].

3-D Mapping and Optimal Coverage

The platforms and the algorithms described in the previous sections were used to implement an autonomous-navigation

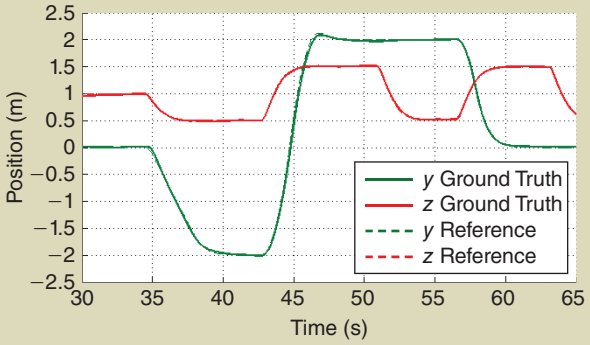

Figure 11. The plot shows the commanded reference trajectory and the measured ground truth.

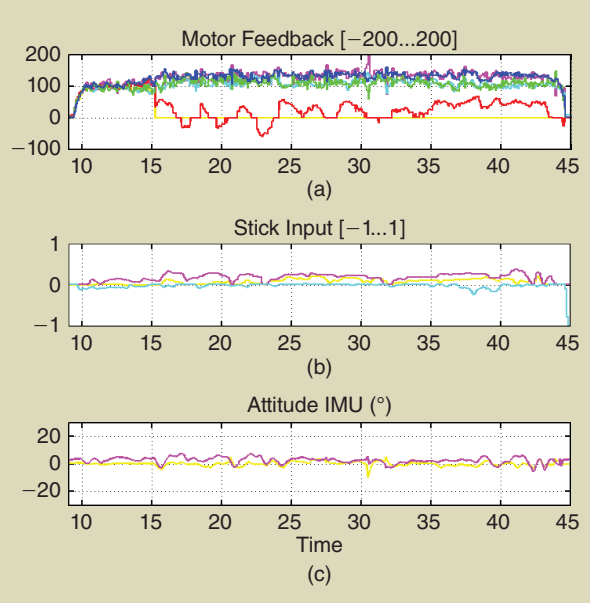

Figure 12. (a) The motor commands in a range [–100, 200] are shown. At about 14 s, the yellow motor is disabled so that the redundancy controller can be activated. As observed, the red motor command slowly oscillates around zero to compensate for the yaw moment. (b) and (c) There is nearly no influence of the failing motor to the pilot's commands or the measured attitude.

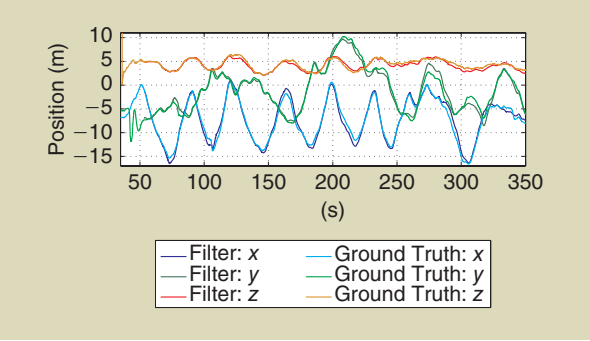

Figure 13. A comparison between EKF-based position estimate (filter: x, y, z) and raw GPS measurements (ground truth: x, y, z) during a 5-min interval of time. The plot suggests that the absolute scale is estimated correctly.

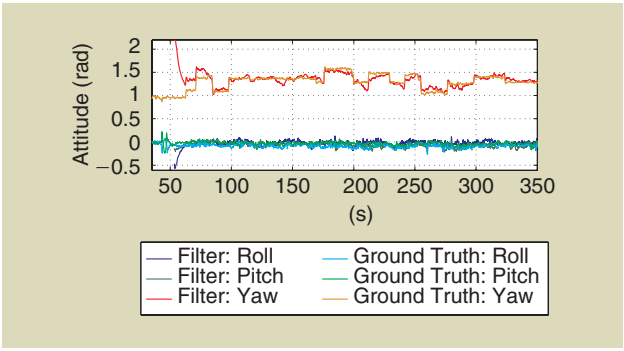

Figure 14. A comparison between EKF-based attitude estimate (filter: roll, pitch, yaw) and GPS-IMU based estimates from the AscTec internal state estimator (ground truth: roll, pitch, yaw) during a 5-min interval of time.

scenario that was publicly demonstrated at the firefighters' training area in Zürich, Switzerland (Figure 16). As described in the "Inertial-Aided Visual Navigation" section, a visual odometry algorithm ran on board each MAV and served for local stabilization as well as for trajectory estimation. At the same time, each MAV built a sparse 3-D map that was incrementally transmitted, together with images and pose estimates, over a Wi-Fi network to a ground-station computer. The ground station, a quad-core Lenovo W520 laptop, was in charge of combining all the received data to compute global position estimates of the three MAVs as well as a dense 3-D map.

Figure 17 shows the pose graphs built by the three MAVs during a flight over the area. These graphs are generated after visual odometry. Drift is visible, especially in the blue trajectory. There, the start and end points are marked with red arrows. The start and end points should overlap in this case, but they do not due to drift. Loop detection, however, recognized the loop closure.

Finally, the three individual submaps are merged into a single global map: first, loop closures are detected between the submaps; then, global bundle adjustment is run over the

Figure 15. After a short initialization phase at the start, vision-based navigation (blue) was switched on for successful completion of a more than 350-m-long trajectory, until battery limitations necessitated landing. The comparison of the estimated trajectory with the GPS ground truth (red) indicates a very low position and yaw drift of the real-time onboard visual odometry.

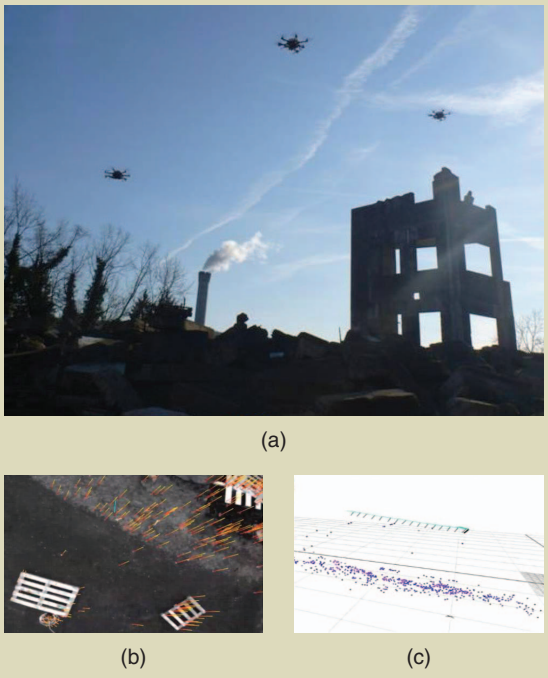

Figure 16. (a) The SFLY helicopters during a demonstration of autonomous exploration at the firefighters' training area in Zürich, (b) feature tracks, and (c) the online-built 3-D sparse map used for local navigation.

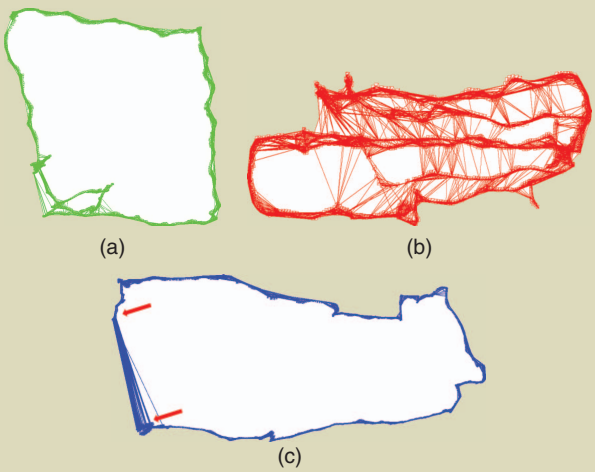

Figure 17. (a)–(c) The pose graphs of three flight trajectories that were used for 3-D mapping. The camera poses are plotted after visual odometry and windowed bundle adjustment. The connecting lines between the cameras show loop closures. As no global optimization is run, pose drift is visible. In (c) the blue trajectory, start and end points are marked with red arrows. The start and end points should overlap in this case, but they do not due to drift. Loop detection, however, recognized the loop closure, and pose-graph optimization will remove the drift (Figure 18).

whole map. Figure 18 shows the pose graph of the final map. The black lines between the cameras of different submaps show the detected loop closures. The global bundle adjustment is able to remove the drift in the individual submaps; and thus, the resulting global map is drift free and in the correct absolute (metric) scale.

A 3-D occupancy map was built, as described in the "3-D Mapping" section. Out of the 3-D occupancy grid, a height map was generated (Figure 19) and fed to the CAO algorithm to compute the optimal-coverage poses. The produced map covers a 42 m × 32 m area with a maximum height of 8.3 m. The final poses for the optimal surveillance coverage of the area by the three MAVs are shown in Figure 20.

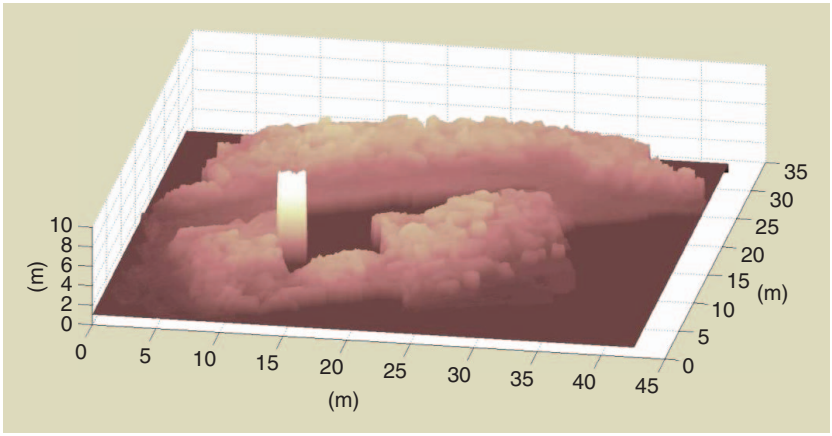

Figure 19. The height map of the Zürich firefighters' training area.

Figure 10 shows a textured visualization of the 3-D environment map of the firefighter area created from three MAVs.

Lessons Learned

Visual–Inertial Sensor Fusion

The flight of more than 350 m outdoors in an unprepared environment (Figure 15) revealed important insights about the system running under real-world conditions. First, the observability analysis of the system, described in the "Local Navigation" section, shows that the system requires excitation to render all states, and, in particular, the visual scale factor, observable. Our tests showed that, under real conditions, this requirement is generally fulfilled. We observed that initializing the visual scale factor correctly (up to about 10% of the true value) is crucial for proper state convergence. In our experiments, we initialized the scale factor either by GPS or by pressure-sensor height measurements. Second, recent work on visual–inertial sensor fusion proposes online calibration of the time offset between the two sensors [44], [45]. While such approaches have high theoretical value, in our experiments, we did not see noticeable differences when increasing or decreasing this offset of maximum 5 ms. This change is significantly larger than the accuracy of common time synchronization protocols like NTP, including jitter on universal serial bus (USB) connections. We estimated once a fixed delay in USB transmissions but did not adapt this estimate during flights or between missions. Third, in the beginning of the project, we experienced significant issues of the visual pipeline [original parallel tracking and mapping algorithm (PTAM)] in self-similar outdoor scenes. Map failures occurred often and marked the end of the mission. Our improvements, described in detail in [59], were key to ensuring continuous operation of the MAV. The most important adaptations include modifying PTAM to a visual odometry framework with constant computational complexity as well as improved feature handling, drastically reducing false positives in the map-building process and

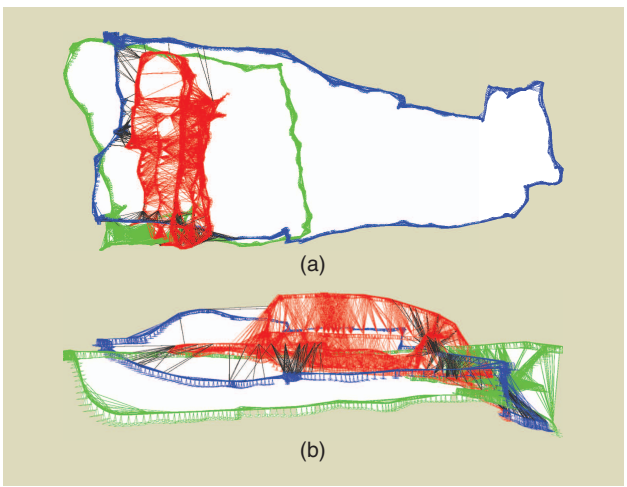

Figure 18. The (a) top and (b) front views of the pose graphs of the three flight trajectories in Figure 17 after map merging and global bundle adjustment. The black lines show the loop closures between the three submaps.

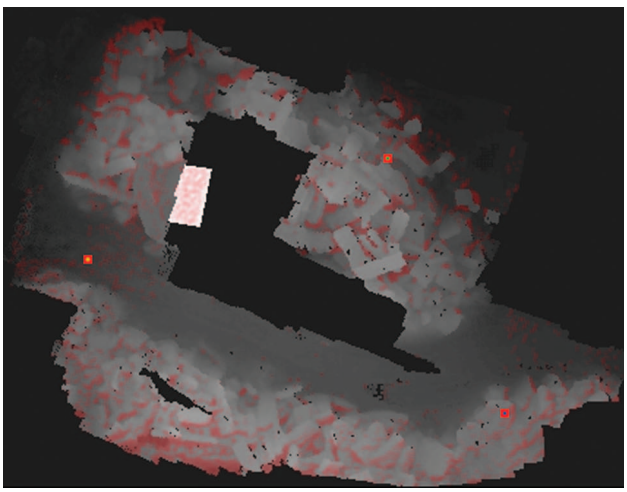

Figure 20. The final configuration of a robot team performing surveillance coverage: the red squares represent the final positions of the MAVs, while the red areas represent the invisible part of the map.

reducing the computational demand at the same time. However, today, more recent VO algorithms, such as SVO [41], represent a more robust, accurate, and faster option for MAVs.

Optimal Coverage

The implementation of the CAO algorithm within the SFLY framework proved the feasibility of an approach relying on an optimization procedure, where the explicit form of the function to be optimized is unknown. A key issue to the successful implementation is the fast and reliable generation of the appropriate inputs to the algorithm. In the case of the SFLY system, the lack of the online map generation resulted in the necessity of offline calculation. The implementation of CAO does not create an approximation or estimation of obstacle location and geometry; instead, it produces on line a local approximation of the unknown cost function that the robots are called to optimize. For this reason, it requires simple and, thus, scalable approximation schemes to be employed, which proved to be ideal for the real-time implementation of CAO.

3-D Mapping

The use of a stereo camera system for 3-D mapping proved to be beneficial. Having a fixed baseline eliminates scale drift in camera pose estimation and makes the dense 3-D reconstruction problem a depth-map fusion problem. Leveraging the IMU measurements for the 3-D reconstruction task proved to be beneficial as well. For feature matching, the relative rotations between two frames is used to predict feature locations and eliminate most of the outliers immediately. This leads to an efficient feature matching and motion estimation step. Our map-merging system is based solely on visual information. Our experiments demonstrated successfully that visual map merging works across multiple platforms even with different camera systems. Performing map merging by pose-graph optimization followed afterward by full bundle adjustment showed to be an efficient way. For dense 3-D reconstruction, we chose to fuse the 3-D measurements into a 3-D grid map prior to digital elevation map generation and triangulation. This avoided problems in 3-D mesh fusion present in other works [60]. Overall, we could successfully demonstrate that it is possible to create large-scale dense 3-D reconstructions using low-weight, low-quality, and low-resolution cameras by fusing a high number of small-scale 3-D reconstructions.

The use of a stereo camera system for 3-D mapping proved to be beneficial.

Conclusions

This article described a framework that allows small-size helicopters to navigate all by themselves using only a single onboard camera and an IMU, without the aid of GPS or active range finders. This framework allows unprecedented MAV navigation autonomy, with flights of more than 350-m length, in previously unexplored environments.

This article shared the experience earned during the three-year European project SFLY about visual–inertial real-time onboard MAV navigation, multirobot 3-D mapping, and optimal surveillance coverage of unknown 3-D terrains. Particular focus was devoted to the technical challenges that have been faced and the results achieved, with detailed insights of how all the modules work and how they have been integrated into the final system. Code, data sets, and videos were made publicly available to the robotics community.

This article highlighted four major contributions of SFLY. The first one is the development of a new six-rotor-based platform robust to single-rotor failures, equipped with enough processing power for onboard computer vision. The second contribution is the development of a local-navigation module based on monocular SLAM that runs in real time on board the MAV. The output of the monocular SLAM is fused with inertial measurements and is used to stabilize and control the MAV locally without any link to a ground station. The third contribution is an offline and offboard dense-mapping process that merges the individual maps of each MAV into a single, global map that serves as input to the global navigation module. Finally, the fourth contribution is a cognitive, adaptive optimization algorithm to compute the positions of the MAVs, which allows the optimal surveillance coverage of the explored area.

To the best of our knowledge, this article describes the first working visual–inertial system of multiple MAVs in real-world scenarios able to autonomously navigate while collaboratively building a rich 3-D map of the environment and performing optimal surveillance coverage. It is believed that the presented system constitutes a milestone for vision-based MAV navigation in large, unknown, and GPS-denied environments, providing a reliable basis for further research toward complete missions of search-and-rescue or inspection scenarios with multiple MAVs.

References

[1] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," in *Proc. SPIE Conf. Unmanned Systems Technology*, 2009, pp. 1–8.

[2] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Auton. Robot.*, vol. 30, no. 1, pp. 73–86, 2010.

[3] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro UAV testbed," *IEEE Robot. Automat. Mag.*, vol. 17, no. 3, pp. 56–65, 2010.

[4] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Proc. IEEE Int. Conf. Robotics Automation*, May 2010, pp. 1642–1648.

[5] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robotics Automation*, 2011, pp. 2520–2525.

[6] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Design and control of an indoor micro quadrotor," in *Proc. IEEE Int. Conf. Robotics Automation*, 2004, pp. 4393–4398.

[7] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles-modeling, estimation, and control of quadrotor," *IEEE Robot. Automat. Mag.*, vol. 19, no. 3, pp. 20–32, 2012.

[8] N. Michael, D. Scaramuzza, and V. Kumar, "Special issue on micro-UAV perception and control," *Auton. Robot.*, vol. 23, nos. 1–2, pp. 1–3, 2012.

[9] M. Cutler, N. Ure, B. Michini, and J. P. How. (Aug. 2011). Comparison of fixed and variable pitch actuators for agile quadrotors. presented at AIAA Guidance, Navigation, Control Conf., Portland, OR. [Online]. Available: http://acl.mit.edu/papers/GNC11 Cutler uber.pdf

[10] M. C. Achtelik, J. Stumpf, D. Gurdan, and K.-M. Doth, "Design of a flexible high performance quadcopter platform breaking the MAV endurance record with laser power beaming," in *Proc. IEEE Int. Conf. Intelligent Robots Systems*, 2011, pp. 5166–5172.

[11] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley, the robot that won the DARPA grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.

[12] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unstructured and unknown indoor environments," in *Proc. European Conf. Micro Air Vehicles*, 2009, pp. 1–8.

[13] A. Chambers, S. Achar, S. Nuske, J. Rehder, B. Kitt, L. Chamberlain, J. Haines, S. Scherer, and S. Singh, "Perception for a river mapping robot," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems*, 2011, pp. 227–234.

[14] S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 90–100, 2012.

[15] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Proc. IEEE Int. Conf. Robotics Automation*, 2011, pp. 20–25.

[16] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3D exploration with a micro-aerial vehicle," in *Proc. IEEE Int. Conf. Robotics Automation*, 2012, pp. 20–25.

[17] D. Floreano, J. Zufferey, M. Srinivasan, and C. Ellington, *Flying Insects and Robots*. Berlin Heidelberg, Germany: Springer-Verlag, 2010.

[18] 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 Int. Conf. Intelligent Robots Systems*, 2005, pp. 3309–3316.

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

[20] S. Ahrens, D. Levine, G. Andrews, and J. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," in *Proc. Int. Conf. Robotics Automation*, 2009, pp. 2643–2648.

[21] M. Bloesch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. IEEE Int. Conf. Robotics Automation*, 2010, pp. 21–28.

[22] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments," *J. Field Robot.*, vol. 28, no. 6, pp. 854–874, 2011.

[23] D. Eberli, D. Scaramuzza, S. Weiss, and R. Siegwart, "Vision based position control for MAVs using one single circular landmark," *J. Intell. Robot. Syst.*, vol. 61, nos. 1–4, pp. 495–512, 2011.

[24] S. Weiss, M. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *Proc. IEEE Int. Conf. Robotics Automation*, 2012, pp. 957–964.

[25] A. Ganguli, J. Cortés, and F. Bullo, "Maximizing visibility in nonconvex polygons: Nonsmooth analysis and gradient algorithm design," in *Proc. American Control Conf.*, 2005, vol. 2, pp. 792–797.

[26] A. Ganguli, J. Cortés, and F. Bullo, "Visibility-based multi-agent deployment in orthogonal environments," in *Proc. American Control Conf.*, 2007, pp. 3426–3431.

[27] P. Agarwal and M. Sharir, "Efficient algorithms for geometric optimization," *ACM Comput. Surv.*, vol. 30, no. 4, pp. 412–458, 1998.

[28] T. Shermer, "Recent results in art galleries," *Proc. IEEE*, vol. 80, no. 9, pp. 1384–1399, 1992.

[29] A. Howard, M. Matarić, and G. Sukhatme, "Distributed coverage control on surfaces in 3D space," in *Proc. IEEE Int. Conf. Robotics Intelligent System*, Lausanne, Switzerland, 2002, pp. 2849–2854.

[30] M. Schwager, B. Julian, and D. Rus, "Optimal coverage for multiple hovering robots with downward facing camera," in *Proc. IEEE Int. Conf. Robotics Automation*, Kobe, Japan, 2009, pp. 3515–3522.

[31] J. Cortés, S. Martínez, T. Karataş, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Automat.*, vol. 20, no. 2, pp. 243–255, 2004.

[32] M. Schwager, J. McLurkin, and D. Rus, "Distributed coverage control with sensory feedback for networked robots," in *Proc. Robotics: Science Systems*, Philadelphia, PA, 2006.

[33] A. Breitenmoser, M. Schwager, J. Metzger, R. Siegwart, and D. Rus, "Voronoi coverage of non-convex environments with a group of networked robots," in *Proc. IEEE Int. Conf. Robotics Automation*, Anchorage, AK, 2010, pp. 4982–4989.

[34] L. Pimenta, V. Kumar, R. Mesquita, and G. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *Proc. IEEE 47th Conf. Decision Control*, Cancun, Mexico, 2008, pp. 3947–3952.

[35] A. Howard, M. Matarić, and G. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in *Proc. 6th Int. Conf. Distributed Autonomous Robotic System*, 2002, pp. 299–308.

[36] A. Breitenmoser, J. Metzger, R. Siegwart, and D. Rus, "Distributed coverage control on surfaces in 3D space," in *Proc. IEEE Int. Conf. Robotics Intelligent System*, 2010, pp. 1–8.

[37] M. Achtelik, K.-M. Doth, D. Gurdan, and J. Stumpf, "Multi-rotor-mavs—A trade off between efficiency, dynamics and redundancy," in *Proc. Guidance, Navigation, Control*, Aug. 2012, pp. 1–8.

[38] D. Scaramuzza and F. Fraundorfer, "Visual odometry: Part I—The first 30 years and fundamentals," *IEEE Robot. Automat. Mag.*, vol. 18, no. 4, pp. 80–92, 2011.

[39] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part II—Matching, robustness, and applications," *IEEE Robot. Automat. Mag.*, vol. 19, no. 2, pp. 78–90, 2012.

[40] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Int. Symp. Mixed Augmented Reality*, 2007, pp. 1–10.

[41] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robotics Automation*, 2014, pp. 1–8.

[42] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *Proc. IEEE Int. Conf. Robotics Automation*, 2011, pp. 4531–4537.

[43] S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV," in *Proc. IEEE Int. Conf. Robotics Automation*, 2012, pp. 31–38.

[44] A. Martinelli, "Vision and IMU data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 44–60, 2012.

[45] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. J. Robot. Res.*, vol. 30, no. 1, pp. 56–79, 2011.

[46] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular SLAM with multiple micro aerial vehicles," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots a Systems*, 2013, pp. 3962–3970.

[47] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, New York, 2006, pp. 2161–2168.

[48] F. Fraundorfer, C. Engels, and D. Nister, "Topological mapping, localization and navigation using image collections," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems*, 2007, pp. 3872–3877.

[49] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. European Conf. Computer Vision*, 2006, pp. 404–417.

[50] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, 2011, pp. 2969–2976.

[51] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robotics Automation*, 2011, pp. 3607–3613.

[52] L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys, "Real-time photorealistic 3D mapping for micro aerial vehicles," in *Proc. IEEE Int. Conf. Robotics Intelligent System*, 2011, pp. 4012–4019.

[53] E. Kosmatopoulos and A. Kouvelas, "Large-scale nonlinear control system fine-tuning through learning," *IEEE Trans. Neural Netw.*, vol. 20, no. 6, pp. 1009–1023, 2009.

[54] J. Spall, *Introduction to Stochastic Search and Optimization*. New York: Wiley, 2003.

[55] J. L. Ny and G. Pappas, "Sensor-based robot deployment algorithms," in *Proc. 49th IEEE Conf. Decision Control*, Atlanta, GA, 2010, pp. 5486–5492.

[56] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos, "Multi-robot three-dimensional coverage of unknown areas," *Int. J. Robot. Res.*, vol. 31, no. 6, pp. 738–752, 2012.

[57] L. Doitsidis, S. Weiss, A. Renzaglia, M. W. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza, "Optimal surveillance coverage for teams of micro aerial vehicles in GPS-denied environments using onboard vision," *Auton. Robot.*, vol. 33, nos. 1–2, pp. 173–188, 2012.

[58] M. Achtelik, S. Lynen, S. Weiss, L. Kneip, M. Chli, and R. Siegwart, "Visual-inertial SLAM for a small helicopter in large outdoor environments," in *Video Proc. IEEE/RSJ Int. Conf. Intelligent Robots Systems*, 2012, pp. 2651–2652.

[59] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *Proc. IEEE Int. Conf. Robotics Automation*, 2012, pp. 957–964.

[60] M. Pollefeys, D. Nister, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. (2008, July). Detailed real-time urban 3D reconstruction from video. *Int. J. Comput. Vision*. [Online]. *78(2–3)*, pp. 143–167. Available: http://dx.doi.org/10.1007/s11263-007-0086-4

Davide Scaramuzza, University of Zürich, Zürich 8050, Switzerland. E-mail: davide.scaramuzza@ieee.org.

Michael C. Achtelik, Ascending Technologies GmbH, Krailling D82152, Germany. E-mail: michael@achtelik.net, michael.achtelik@asctec.de.

Lefteris Doitsidis, Technological Educational Institute of Crete, Chania 73100, Greece. E-mail: ldoitsidis@chania.teicrete.gr.

Friedrich Fraundorfer, Technische Universität München, München 80333, Germany. E-mail: friedrich.fraundorfer@tum.de.

Elias Kosmatopoulos, Democritus University Thrace and ITI/CERTH, Xanthi 67100, Greece. E-mail: kosmatop@dssl.tuc.gr.

Agostino Martinelli, INRIA Grenoble-Rhone-Alpes, Grenoble 38334, France. E-mail: agostino.martinelli@inria.fr.

Markus W. Achtelik, Autonomous Systems Lab, ETH Zürich, Zürich 8092, Switzerland. E-mail: markus.achtelik@mavt.ethz.ch.

Margarita Chli, University of Edinburgh, Edinburgh EH8 9AB, United Kingdom. E-mail: margarita.chli@mavt.ethz.ch.

Savvas Chatzichristofis, Democritus University Thrace and ITI/CERTH, Xanthi 67100, Greece. E-mail: schatzic@ee.duth.gr.

Laurent Kneip, Australian National University, Canberra 0200, Australia. E-mail: laurent.kneip@anu.edu.au.

Daniel Gurdan, Ascending Technologies GmbH, Krailling 82152, Germany. E-mail: daniel@gurdan.de.

Lionel Heng, ETH Zürich, Zürich 8092, Switzerland. E-mail: hengli@inf.ethz.ch.

Gim Hee Lee, ETH Zürich, Zürich 8051, Switzerland. E-mail: lgimhee@gmail.com, glee@student.ethz.ch.

Simon Lynen, ETH Zürich, Zürich 8092, Switzerland. E-mail: simon.lynen@mavt.ethz.ch.

Lorenz Meier, ETH Zürich, Zürich 8092, Switzerland. E-mail: lm@inf.ethz.ch.

Marc Pollefeys, ETH Zürich, Zürich 8006, Switzerland. E-mail: marc.pollefeys@inf.ethz.ch.

Alessandro Renzaglia, University of Minnesota, Minnesota 55455, USA. E-mail: a.renzaglia@gmail.com, arenzagl@umn.edu.

Roland Siegwart, ETH Zürich, Zürich 8092, Switzerland. E-mail: rsiegwart@ethz.ch.

Jan Carsten Stumpf, Ascending Technologies GmbH, Krailling 82152, Germany. E-mail: jan@asctec.de.

Petri Tanskanen, ETH Zürich, Zürich 8092, Switzerland. E-mail: tpetri@student.ethz.ch.

Chiara Troiani, INRIA Rhone Alpes, Grenoble 38334, France. E-mail: chiara.troiani@inrialpes.fr, t.chiara@gmail.com.

Stephan Weiss, NASA Jet Propulsion Laboratory and California Institute of Technology, Pasadena 91109, California, USA. E-mail: stephan.weiss@ieee.org.