By Teodor Tomić, Korbinian Schmid, Philipp Lutz, Andreas Dömel, Michael Kassecker, Elmar Mair, Iris Lynne Grixa, Felix Ruess, Michael Suppa, and Darius Burschka

# Toward a Fully Autonomous UAV

## *Research Platform for Indoor and Outdoor Urban Search and Rescue*

©PHILIPP LUTZ

Urban search and rescue missions raise special requirements on robotic systems. Small aerial systems provide essential support to human task forces in situation assessment and surveillance. As external infrastructure for navigation and communication is usually not available, robotic systems must be able to operate autonomously. A limited payload of small aerial systems poses a great challenge to the system design. The optimal trade-off between flight performance, sensors, and computing resources has to be found. Communication to external computers cannot be guaranteed; therefore, all

*Digital Object Identifier 10.1109/MRA.2012.2206473*
*Date of publication: 28 August 2012*


1070-9932/12/$31.00©2012IEEE

processing and decision making has to be done on board. In this article, we present an unmanned aircraft system design fulfilling these requirements. The components of our system are structured into groups to encapsulate their functionality and interfaces. We use both laser and stereo vision odometry to enable seamless indoor and outdoor navigation. The odometry is fused with an inertial measurement unit in an extended Kalman filter. Navigation is supported by a module that recognizes known objects in the environment. A distributed computation approach is adopted to address the computational requirements of the used algorithms. The capabilities of the system are validated in flight experiments, using a quadrotor.

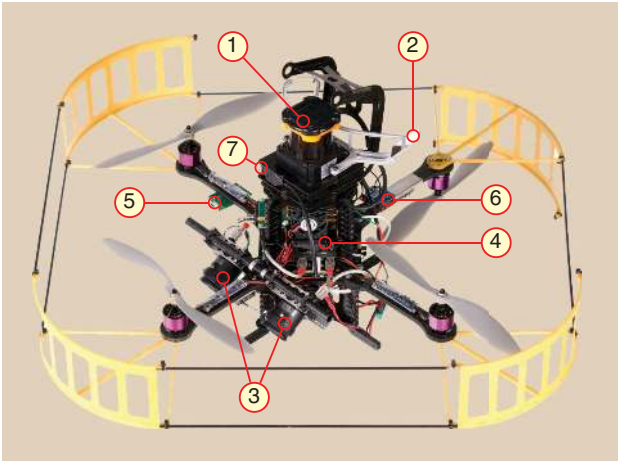
Civil and commercially oriented unmanned aerial vehicle (UAV) missions range from rather modestly structured tasks, such as remote sensing (e.g., wild-fire detection), to highly complex problems including common security jobs or search and rescue (SAR) missions. Disaster search and urban rescue-related missions are still a fairly demanding challenge because of their exceedingly variable nature. The mission planning has to take a multitude of scenarios into account, considering arbitrary, unknown environments and weather conditions. It becomes apparent that it is also not feasible to preconceive the large number of unforeseeable events possibly occurring during the mission. In research, certain types of SAR, in particular, wilderness SAR [1], [2], have already been successfully mastered using UAV systems. However, to date, the persistent performance of robotic systems operating in an urban environment is very challenging, even with a low degree of human intervention [3]. A stable broadband radio link cannot be guaranteed in such environments because it requires a high level of autonomy of the systems. The limited availability of computing resources and low-weight sensors operating in harsh environments for mobile systems pose a great challenge in achieving autonomy.

Our research goal is to develop robotic systems capable of accomplishing a variety of mixed-initiative missions fully autonomously. Completely autonomous execution of urban SAR (USAR) missions poses requirements on the robotic systems operating therein. Various such missions require the robots to be modular and flexible in terms of sensor and planning capabilities. The robots have to operate in unstructured indoor and outdoor environments, such as collapsed buildings or gorges. Navigation systems therefore have to work without external aids, such as global positioning systems (GPS), since their availability cannot be guaranteed. Flying systems additionally have to provide robust flight capabilities because of the changing local wind conditions in such environments. A key feature to achieving full autonomy in urban disaster areas is on-board processing and decision making. Search assignments also require mission-specific recognition capabilities on the robots.

As a first step, we have developed a modular and extensible software framework for autonomous UAVs operating in USAR missions. The framework enables a parallel and independent development of modules that address individual challenges of such missions. It features reliable flight and navigational behavior in outdoor and indoor environments, and permits execution of higher level functions such as the perception of objects and persons, failsafe operation, and online mission planning. The framework is implemented and tested on a commercial quadrotor platform. A quadrotor has been chosen because of its favorable rotor size and safety when compared to a conventional small-size helicopter. The platform has been extended in terms of sensor, computer, and communication hardware (Figure 1).

Similar platforms have already been developed by other researchers. The platforms are tailored to solve a simultaneous localization and mapping (SLAM) problem. This problem requires a lot of computational power, which is not readily available on flying systems. Therefore, the authors in [4] and [5] chose to send laser scanner data to a ground station for processing. Pose estimation and high-level tasks are done on the ground station, whereas control and stabilization of the platform are done on the quadrotor. More recently, through the optimization of algorithms and faster processors, pose estimation and planning have been done onboard. Notable implementations are laser based [6] for indoor environments and monocular visual SLAM [7] for both indoor and outdoor environments. The pose estimate of the SLAM solution is commonly fused with inertial measurement unit (IMU) measurements in an extended Kalman filter (EKF) to obtain a full-state estimate, which is then used for control.

Our approach differs from the previous work in three major ways. First, instead of one sensor, we rely on two complementary exteroceptive sensors. This enables flight in both indoor and outdoor environments. As in state-of-the-art systems, the respective odometry is fused with the IMU using an EKF. Second, no geometric map is built. Instead, we correct for drift errors by recognizing known landmarks in




PHILIPP LUTZ

**Figure 1.** An experimental platform based on the Ascending Technologies Pelican quadrotor, showing 1) laser scanner, 2) mirrors, 3) stereo cameras, 4) a modular computation stack, 5) wired Ethernet connection, 6) XBee modem, and 7) WLAN stick. One of the propellers is pointing downward to improve the view of a front-facing camera (not depicted).

the environment. This lends itself to navigation in larger environments, as memory requirements are smaller when compared to SLAM. To guarantee our robots' autonomy, all processing is done onboard, akin to most recent systems. There are many computationally intensive tasks, which need to run simultaneously—stereo processing, visual odometry, laser odometry, and computer vision. In contrast to the state-of-the-art approaches, we adopt a distributed computation platform consisting of several onboard boards instead of one.

The components of our framework are inspired by the international micro air vehicle (IMAV) [8] indoor exploration challenge. The objective of the challenge is to fly into a house of known shape and dimensions, detect objects, and return outside to land on a defined pattern. Several problems found in USAR missions have to be addressed. The UAV system must first find the house, as it starts behind a wall without direct sight of the house. Once found, precise detection of and navigation through the door, window, or chimney of the house is required. Pattern recognition is used for object detection and landing zone identification. External aids, such as GPS or a motion tracking system, are not available. Adding artificial features to the environment is penalized. Although not all difficulties of a USAR mission are addressed, the navigation, autonomous decision making, and object recognition challenges are present. The size of our 70-cm-wide platform in relation to the 1-m-wide passages into the house poses an additional challenge. Our system's architecture is explained in terms of the aforementioned mission.

Current approaches, which try to tackle this kind of challenge, use a laser scanner [5] or monocular vision [9]. The processing in these approaches was done offboard. For the vision system, artificial features had to be added to the indoor environment. Our system will use the best odometry sensor in a given situation. Systems have autonomously flown into the house through the window and doors; however, no system has yet flown the complete mission autonomously.

## Software Framework

Our modular framework consists of intercommunicating components, enabling the easy exchange of task-related functionality and exchangeability of components. To further define their scope, the system components are subdivided considering their degree of autonomy and cognitive functionality, as depicted in Figure 2.

Concerning the level of autonomy, the system is structured into low- and high-level components. The low-level components are responsible for the data fusion and flight control of the quadrotor. They allow for reliable autonomous flight and navigation, shared through an abstract and unified interface between humans and high-level components. As the stability of the system depends on these components, a hard real-time system with a high execution rate is required. The high-level components provide situational awareness and mission planning functionality with a representation of the environment. The status of the system, the mission, and the environment are monitored, and commands are issued accordingly. They take over tasks usually done by a human operator.

Furthermore, the components are grouped into perception, cognition, and action layers [10], [11] as depicted in Figure 2. The perception layer includes all tasks concerning acquisition and processing of sensor data. Therein, the data

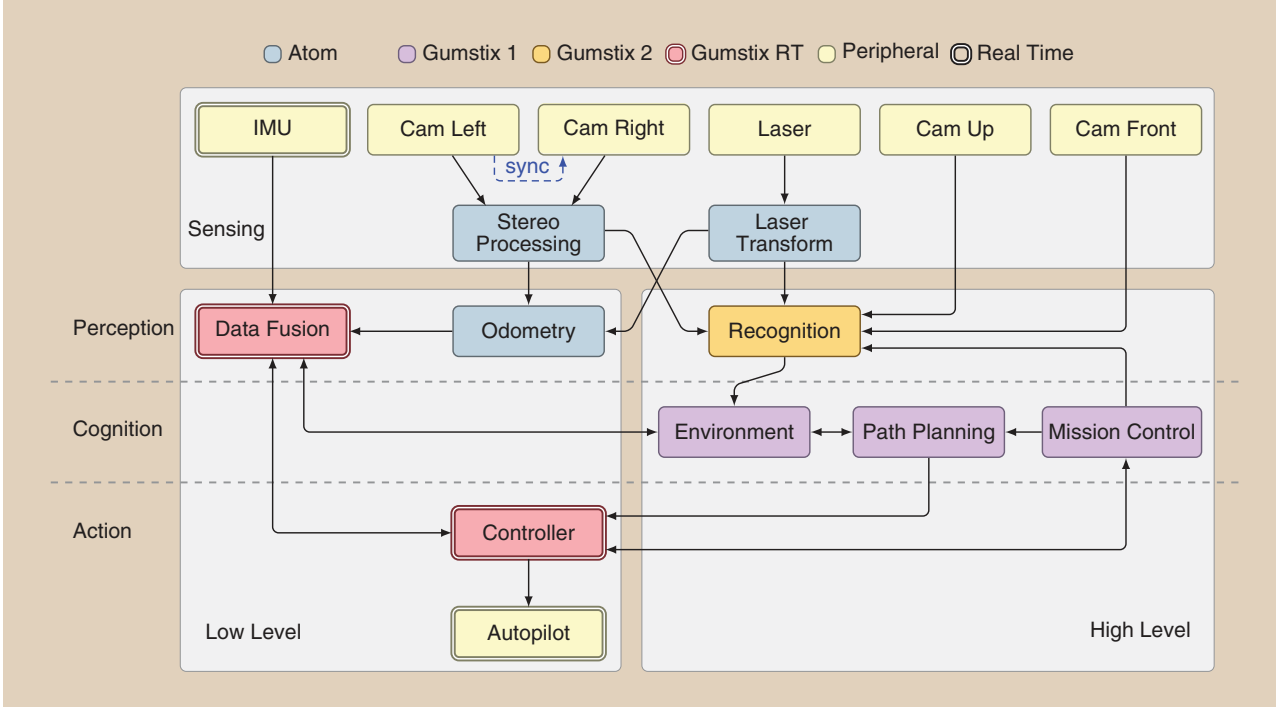



**Figure 2.** System architecture and software deployment of components. In addition to being organized into cognitive layers, the system components are partitioned according to their autonomy level as well as cognitive functionality.

fusion component fuses the proprioceptive and exteroceptive sensor data. Mission-dependent recognition of world features, such as persons or interesting objects, is done in the recognition module. World representation, as well as planning and decision-making functionalities, is realized in the cognition layer. Lastly, the action layer is involved in stabilizing and moving the UAV in the desired manner. Such categorization allows for a clear definition of the interfaces between components and the minimum required set of component functionalities. The realization of this structure in our experimental system can be presented more clearly when grouped by the layer subdivision.

## Perception

The UAV should be able to fly in structured indoor environments as well as outdoors. The indoor environments consist of clearly defined vertical structures (walls) that can be detected by a laser scanner. However, poor lighting conditions and a low number of environment features make indoor environments unsuitable for a camera-based odometry system. Conversely, the outdoor environments lack clear structures. Sunlit environments contain light in the part of the spectrum that coincides with that used by infrared laser scanners, disturbing the measurements. This makes low-powered lightweight laser scanners, which are commonly employed on flying systems, unsuitable for such environments. Outdoor environments have many natural features and good lighting conditions, which makes them well suited for visual odometry systems. In such environments, previous camera images can be easily recognized, so the camera can be used for loop closure.

In our approach, we use both laser and stereo odometry for pose estimation. The combination of two odometry approaches allows compensating drawbacks of a single sensor. Moreover, the estimation of all six degrees of freedom (6 DoF) states can be done using only one filter. This differs from other approaches, where either laser odometry [6] or monocular visual odometry [7] is used for pose estimation.

The stereo camera in our system points downward not only to ensure that the odometry is available in outdoor areas but also to enable detection of a target from above. Drift errors can be compensated by using key frames in the visual odometry system, as well as recognition of known landmarks in a topological map. For the indoor exploration mission, the map is fixed and predefined, as known landmarks include the window, door, and chimney. Their exact position is known with respect to the house, so they can be used to correct drift errors. These are detected and tracked using front-facing and upward-facing cameras (not shown in Figure 1), respectively. Two separate cameras provide more stable tracking than a pan-tilt unit with one camera of the same weight.

## Odometry

### *Laser Odometry*

The laser odometry system is based on Censi's canonical scan matcher [12]. The laser scan is projected to the ground plane in the laser transform component, using attitude information from the data fusion component (Figure 2). The projected data are only valid for scan matching if the scanned environment objects contain vertical planes. This assumption is valid for most indoor environments. The algorithm uses an iterative closest point (ICP) variant to compute three-dimensional (3-D) delta movement information [change in (x, y) position and yaw angle] between two points in time and the corresponding measurement covariance.

### *Visual Odometry*

A correlation-based algorithm [13], [14] is used to obtain a disparity image from two time-synchronized camera images in the stereo-processing component. Based on this 3-D information, the six-dimensional delta position and orientation between two points in time as well as the corresponding measurement covariance are calculated [15]. The algorithm supports a key frame buffer so that the delta measurement refers not just to the last acquired image but also to the image in the buffer that gives the delta measurement with the smallest absolute covariance.

As shown in the "Experimental Results" section, the estimated variances for laser and camera odometry are a good indicator to classify the environment into indoor and outdoor. In the variance calculation for each sensor, it is assumed that there are no outliers in the measurement. During the experiments, we have found that, under bad sensor conditions, outliers in the measurements occurred. These could not be detected by an outlier rejection mechanism using Mahalanobis distance. Therefore, the measurement variance is invalid. Fusing these measurements would lead to unpredictable behavior of the filter. Because of this, we switch to the sensor that works well in a specific environment. We assume that the sensor with the smallest measurement variance is best suited in the current environment and is therefore used for fusion.

## Data Fusion

The proprioceptive sensor information from the IMU and the exteroceptive odometry information have to be fused to get the current system state estimate. There are two main challenges.

First, the odometry data give only relative position and orientation information. Second, the odometry data are time delayed because of measurement and data processing time. Precise times of measurement are obtained through hardware synchronization triggers. The total delay of the laser odometry in the experimental system is about 100 ms with an update frequency of 10 Hz, and for the visual odometry, the delay is more than 300 ms, with a frequency of 3 Hz. Therefore, the measurement refers to a state in the past. As the estimate is used to control the UAV, and the quadrotor dynamics are fast compared to the measurement delays, the latter have to be considered in the data fusion algorithm. This is realized using an indirect feedback Kalman filter with state augmentation [16] using two state vectors.

The direct state vector includes position, velocity, and attitude (as quaternion) in the world frame and the biases of the acceleration and gyro sensors in the body frame $\in \mathbb{R}^{16}$. Quaternions are used to circumvent the gimbal lock problem that might occur when using minimal attitude representations. The direct state is calculated by the strapdown algorithm.

The main filter state vector includes the errors in position, velocity, and orientation in the world frame and the IMU acceleration and gyro bias errors in the system body frame $\in \mathbb{R}^{15}$. Since we assume small errors in the filter, the small angle approximation is employed to efficiently represent the attitude. Hence, the scalar part of the quaternion can be omitted, as it is implicitly defined to one. This also simplifies modeling of the attitude sensor noise. A hardware synchronization signal of the laser and the camera system signaling the start of a data acquisition sequence is directly registered by the real-time system running the filter algorithm. At every synchronization trigger, a substate including position and orientation of the current direct state is saved and augmented to the main filter state. The delayed delta measurement includes two time stamps for each measurement. These time stamps are used to find the corresponding states within the state vector and construct a suitable measurement matrix referencing the selected states.

The absolute position and orientation of the system are unobservable with only delta measurements, which are reflected in an unbounded covariance for these states. Therefore, further absolute measurements are included:

- The height to ground is measured by laser beams reflected to the ground. Height jumps caused by objects lying on the ground are detected and compensated.
- Measurement of the gravity vector is used as pseudoabsolute measurement for roll and pitch.
- Measurements with respect to known landmarks, if available, are used to correct position drift errors.

If absolute position measurements arrive only in the range of minutes, there might be small jumps in the position estimate. Nevertheless, these jumps do not cause jumps in the velocity estimate as its covariance is bounded by the regular delta position measurements. This is an important feature for the underlying UAV controller, as jumps in the velocity prediction can significantly degrade flight performance.

### Recognition

Identifying and locating persons, animals, or objects (e.g., landmarks, signs, or a landing zone) is a central issue in USAR missions. The conceptual idea behind the recognition module is to offer related object detection and recognition services. The module acts as an interface between the mission planner, environment (cognition), and the sensors. Triggered by the mission planner, it interprets sensory information and returns semantic and location information, respectively. The recognition module supports absolute localization in the sense that it detects known objects and estimates their relative positions and heading with respect to the UAV frame. It leverages typical object recognition techniques in computer vision and 3-D point-cloud processing. Three demonstrator recognizers are currently implemented: a pattern recognizer for two-dimensional images, a house detector based on stereo vision, and a laser object detector.

The pattern recognizer is typically used when searching a marked landing zone. The pattern is a gray-value image or drawing of the landing zone. Together with a description of its size, the pattern matcher checks for similar occurrences in camera images in a more efficient way than a common template matching approach. The first steps try to reduce the problem size by segmenting the image data. A corner detector is applied to the template image to obtain interesting points. Small patches are then generated around these points and stored in a database. A simple operation to calculate a descriptor of the patches based on orientation histograms is used [17]. Subsequently, the descriptors are compared pairwise using normalized cross-correlation, sorted in an arbitrary number of classes, and a Bayes classifier is learned. This results in a sequence of descriptors, which in turn define the pattern.

Specifically for the IMAV challenge, we have developed a house detector and laser object detector. The house detector uses disparity images. It is used to detect the house from above, since the cameras are pointing downward. A combination of principal component analysis and an algorithm similar to the ICP algorithm is used to fit the shape of a model of the house into the point cloud. In addition, parts of the house (e.g., chimney) are identified in a monocular image of the stereo pair and fused with results of the point cloud fitting. The laser object detector is able to detect corners in a room, walls, and windows.

### *Action*

The controller component implements a position controller running at 100 Hz on the real-time system. Control inputs are attitude commands that are sent to the autopilot, which implements a PD attitude controller in a 1-kHz control loop. The purpose of the position controller is to follow a reference position, velocity, and acceleration, using the data fusion's pose estimate. The position controller is a full-state feedback controller that uses a combination of integral sliding mode [18] and time-delay disturbance estimation [19]. The integral action provides a zero steady-state error, whereas the disturbance estimator uses accelerometer measurements and previous control inputs to respond to disturbances faster. This combination provides sufficient robustness to fly in indoor and outdoor environments and through narrow passages.

In-flight switching and configuration of position controller implementations simplify their testing. The position between two waypoints is interpolated as a straight line in Cartesian space using a constant velocity. This interpolated position is run through a linear filter that represents the quadrotor's translational dynamics to generate smooth

reference trajectories. Using this method, it is easy to configure the transient behavior of the vehicle's position by setting the interpolation velocity and filter parameters. Such configurations are stored as flight modes (e.g., fast, careful, accurate by decreasing velocity). A unified interface allows flight modes to be set for each path segment individually. High-level components can set the flight mode according to the current mission task.

A state machine in the low-level system is implemented in the controller as depicted in Figure 3 and defines the activation of components based on the readiness level of the system, as summarized in Table 1. It signals the availability of low-level abilities to high-level components. The system starts in the preparation state, during which data fusion and position control are disabled, so the quadrotor can be moved freely to a starting position. This is useful for initialization of the system before a mission, as the movement will not affect the state estimate. It is assumed that the quadrotor is stationary in the powered off state, so the data fusion is initialized and state estimation starts. Upon engaging the motors, the system is in the powered on state and the sensor data fusion assumes that the quadrotor is moving. The initial state therein is landed, which assumes that the quadrotor is still on the ground. The take-off command activates the position control feedback loop, while commanding the quadrotor to hover at a predefined height above the starting location. During the ascend, the system is in the transitional taking off state, which can be canceled with the land command. Once the hover point is reached with defined accuracy, the system is automatically transitioned into the nominal fault-free flying state, and paths can only be flown in this state. Landing occurs through the transitional landing state analogously to taking off. The transitional states ensure that corresponding physical changes have been safely completed before allowing any other actions to be taken. This abstraction greatly simplifies experiments, since components are activated and initialized as needed.

To ensure a fall-back strategy in case of fatal errors during flight, a fail-safe state is implemented in which the system performs an emergency landing routine. For example, the system enters this state automatically if a data fusion divergence has been detected. This shall protect the system and minimize the possibility of harm to humans or to the platform itself. In such an event, fused data are not used for position control—instead, only the raw altitude measurement from the laser scanner is used for descending, while the vertical velocity is obtained by using an $\alpha$–$\beta$ filter of the measured altitude. Altitude stabilization is active through the autopilot and does not depend on the fusion information. In this state, the quadrotor's horizontal position will drift, but more importantly, the quadrotor will not crash to the ground, and it is easier for a safety pilot to take over. This method of descent is safer than simply reducing the thrust or turning the motors off. It has proven to be useful during experiments and when testing new components.

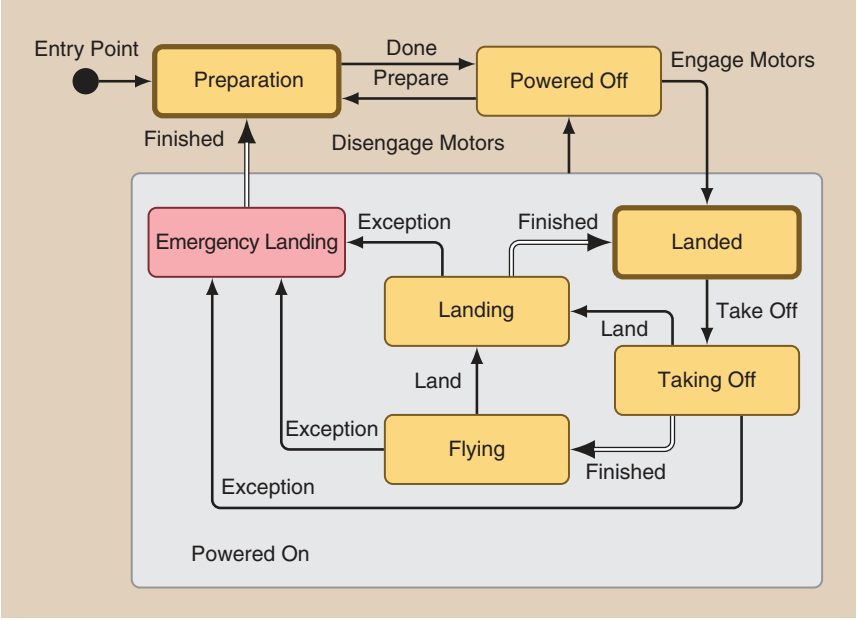



**Figure 3.** Low-level state machine implemented in the controller component.

## Cognition

Completely autonomous execution of USAR missions requires interpretation of the robot's environment and the performing of actions accordingly. To achieve that, the robot requires a representation of the world, as well as path-planning and decision-making capability. In our framework, these are implemented in the cognition layer. Its modularity allows for implementation of different algorithms through the use of the available interfaces and choosing the combination best suited for a particular mission.

**Table 1. Activation of system components depending on low-level system state.**

| System State | Fusion | Control | Waypoints |
|---|---|---|---|
| Preparation | — | — | — |
| Powered off | • | — | — |
| Landed | • | — | — |
| Taking off | • | • | — |
| Flying | • | • | • |
| Landing | • | • | — |
| Emergency landing | — | • | — |

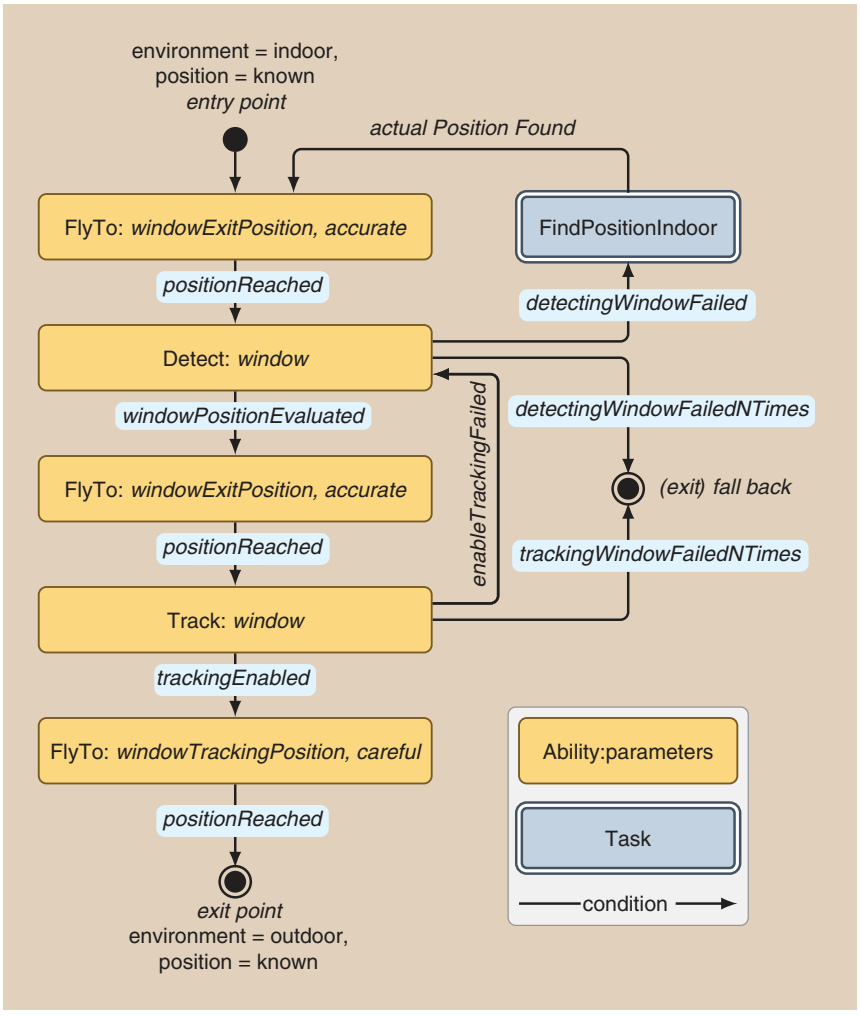



**Figure 4.** Example of a mission control task–leave house by window. The task is a state machine composed of atomic states called abilities. Tasks are fixed and mission specific.

We will illustrate this by the leave house by window task, depicted in Figure 4. The quadrotor first flies to a window position that is stored in a map. Once this position is reached, a window detector tries to determine the window position more precisely using vision and thereafter slowly approaches it. If successful, the UAV flies to the determined position.

At this point, the visual servoing starts. The position determined by the window tracker is continuously sent to the controller. The UAV slowly flies through the window midpoint while the window is visible. Once the other side of the window is reached, the task is finished. Window detection and tracking services are provided by the recognition module. If the window cannot be detected precisely, the FindPositionIndoor fall-back task is invoked. This determines the quadrotor position in relation to the house. In the case of too many detection or tracking failures, the task exits to a fall-back task, like leaving the house through the door.

## Hardware and Infrastructure

Because of high payload capacity, the Ascending Technologies Pelican quadrotor was chosen as the flight platform, which is shown in Figure 1. With a total weight of 2.05 kg, our system hovers at approximately 70% of the quadrotor's maximum thrust, leaving a control reserve that is only sufficient for relatively slow maneuvers. Maximum flight time is approximately 10 min with one accumulator. The used hardware components are listed in Table 2. The most notable difference to similar systems is the time-synchronized modular computation stack, connected through Ethernet.

A Hokuyo UTM-30LX laser scanner and PointGrey Firefly cameras are used as exteroceptive sensors via USB, connected to different computers to parallelize the data acquisition process.

The onboard computational hardware consists of one CoreExpress Atom board, three Gumstix Overo Tide boards, and an Ethernet switch, as shown in Figure 5. The atom board is used for stereo processing because of high computational requirements. Image processing and cognition tasks are executed on dedicated Gumstix boards. If more computational power is required, computers can be added to the system without changes in the system architecture.

For the exploration challenge, the current implementation of the environment component contains a world model and objects therein as a topological map. If a known object is detected with a high confidence by the recognition component, drift errors can be corrected by sending the relative location of the object to the data fusion. The mission control component provides autonomous mission execution through a hierarchical state machine composed of tasks. The abilities are atomic states of each task, and they represent basic functionalities or actions provided by other system components and can be invoked with parameters.

For example, the FlyTo ability invokes the path planning component, which uses environment information to find a list of waypoints from the current position to the desired topological pose while avoiding obstacles and dangerous zones. Together with the specified flight mode, the determined path is then sent to the controller for execution, i.e., the plan is static while the ability is active. Once the last waypoint is reached with sufficient accuracy, a transition is triggered.

The time-critical strapdown, data fusion, and control tasks run on the real-time system. It is an Ubuntu Linux with an RT-patched kernel and is connected to the autopilot, which also provides the IMU. A high IMU poll rate is required for the strapdown algorithm; therefore, $I^2C$ communication with the autopilot runs at 400 kHz. The resulting bandwidth allows IMU data polling at 200 Hz and attitude command sending from the position controller at 100 Hz. All remaining Gumstix computers are running latency-tolerant high-level tasks such as image processing and mission control on an Ubuntu Linux operating system.

An eight-port fast-Ethernet switch is used for high-bandwidth onboard communication between the computing hardware, shown in Figure 6. Ethernet has been chosen because of well-established standard protocols, low-latency communication, and readily available middleware. Screwable GIGCON connectors provide vibration-resistant connections of the Ethernet cables. An external data connection to the system is possible through wired Ethernet, WLAN, XBee modem, and USB. WLAN is made possible through a tiny USB stick connected to the Atom computer with a maximum bandwidth of 150 Mb/s. The Ubuntu Linux on the Atom processor runs a software bridge where incoming connections from WLAN are routed to the internal onboard network. A slower and more reliable connection is provided by the XBee modem connected to a serial port of the Atom computer. Lastly, each of the Gumstix computers provides a serial terminal interface over USB, used only when the system is not flying.

The distributed approach of splitting tasks among multiple computers requires a suitable middleware to enable communication between them. Our software framework poses the following requirements: scalability and support for distributed nodes; clock synchronization; flexible data formats and API; small footprint (usable for embedded systems); suitability for robotic applications and software. As a result of the evaluation of different frameworks and middleware, robot operating system (ROS) was chosen as most suitable, although it lacks clock synchronization and real-time communication because of its design.

All real-time critical tasks run as threads in a single process (nodelet), so they communicate through shared memory. A good example is the data flow from IMU to data fusion to controller to autopilot as can be seen in Figure 2. This zero copy transport approach is also used to reduce communication overhead where large amounts of sensor data are shared among software modules.

An open-source implementation of PTPd2 (precision time protocol daemon) is used for time synchronization between the computers. Low data bandwidth and a synchronization rate of 4 Hz are sufficient to maintain an average deviation of system clocks well below 500 $\mu$s. The Atom board serves as master clock, and all other computers are configured as slave clocks. On all computers the PTPd daemon runs with a high real-time priority to keep operating system scheduling latency as short as possible.

Deployment of compiled nodes and configuration files is done from external development computers using an enhanced ROS build workflow, which invokes rsync program for fast data synchronization to the research platform over Ethernet or WLAN. ROS launch files are used to run and configure nodes across all computers with only one command.

**Table 2. Hardware components of the experimental system.**

| Component | Hardware |
|---|---|
| Quadrotor | AscTec Pelican |
| Atom board | 1.6 GHz Intel Atom, 1 GB DDR2 |
| Gumstix boards | ARM Cortex-A8, 720 MHz, 512 MB RAM |
| IMU, accelerometer | Memsic MXR9500M |
| IMU, gyroscope | Analog Devices ADXRS610 |
| Laser | Hokuyo UTM-30LX, 30m, 270° scanning range |
| Cameras | PointGrey Firefly FMVU-03MTM/C-CS |
| XBee | XBee 2.4GHz radio modem |
| WLAN | Lightweight USB module, max 150 Mb/s |
| Switch | 100 Mb, eight port switch |

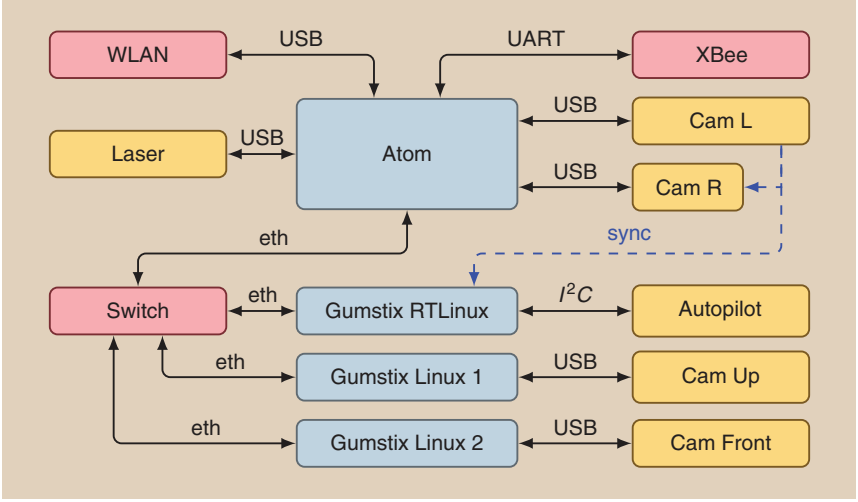
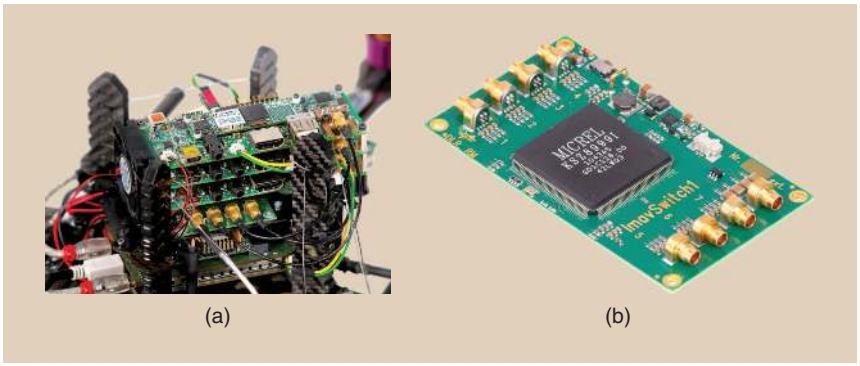



**Figure 5.** Onboard distributed computation architecture and sensor communication. sync: synchronization; eth: Ethernet.




PHILIPP LUTZ

**Figure 6.** Details of avionics hardware components. (a) Modular computation stack. (b) An eight-port fast Ethernet switch.

## Experimental Results

A flight experiment was conducted to show the effectiveness of using two sensing paradigms. Figure 7 shows the estimated path flown inside and outside the experimental facility (depicted in Figure 8). Figure 9 shows the reference and estimated system states as well as absolute covariances of the two odometries. Time instants when the system switches between visual and laser odometry are marked on the time axis. Because of practical difficulties in outdoor measurements, no ground truth is provided. A rectangular path around the house was chosen because it was clear of ground obstacles. The quadrotor has been yawing during the flight so that the laser scan points toward the house.

The quadrotor starts inside the house using laser odometry. Visual odometry is not available because the cameras are too close to the ground, and so there is not enough overlapping in the images. At 7.5 s, during autonomous take-off, when the quadrotor is at 68 cm altitude, the depth image becomes available, and the covariance of the visual odometry becomes smaller than that of laser odometry. Therefore, the system switches to visual odometry.

Shortly afterward, the system is commanded to fly outside. At 21 s, visual odometry becomes unavailable as indicated on the out-of-axis covariance in Figure 9. This is caused at first by motion blur when the quadrotor starts moving in the weak lighting conditions in the house. The system automatically switches to laser odometry. During flight through the 1-m-wide window, a jump in the raw laser height measurement can be seen due to flying above the 20-cm-wide wall. The jump is detected by the data fusion and a constant altitude is kept. The vertical velocity is also unaffected, so the vehicle passes smoothly through the window. The visual odometry is still unavailable as the window pane is too close to the cameras.

When the quadrotor is outside, the cameras need to adjust their exposure time, so visual odometry is again available at approximately 1 m behind the window. It is clearly visible that the covariance of the laser odometry outdoors is very large compared to indoors, due to less valid laser measurements. Therefore, only visual odometry is used for the outdoor flight.

During autonomous landing, the disparity image becomes unavailable under 60 cm of altitude. This is indicated by the high covariance of the visual odometry, and the system switches to laser odometry.

The reference velocity and position are tracked according to the estimated values. The position control error with respect to the estimated states is under 20 cm in both indoor and outdoor environments.

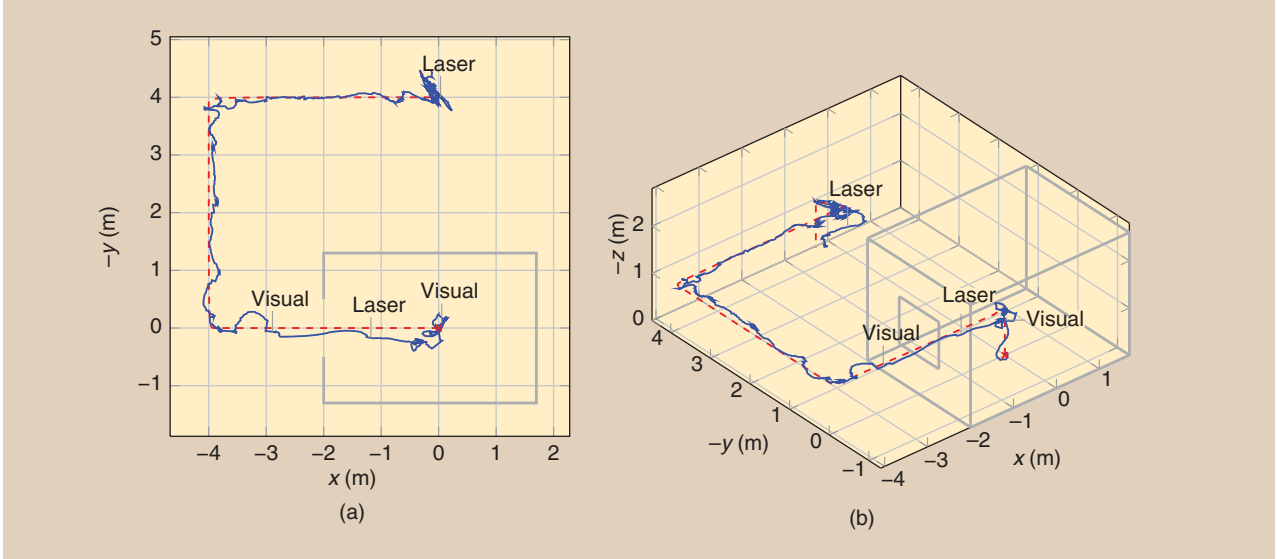



**Figure 7.** Flown path estimated in the experiment. The dashed red line shows the reference trajectory, while the solid line shows the estimated path. The house outline is shown in gray. Locations where switching between visual and laser odometry occurs are also indicated. (a) Estimated flown path in the horizontal ($x$, $y$) plane. (b) Three-dimensional view of the desired and estimated position ($x$, $y$, $z$) of the quadrotor during the experiment.

## Conclusion and Future Work

We introduced a modular and extensible software and hardware framework for the autonomous execution of USAR missions using aerial robots. An implementation of the framework on an experimental quadrotor system has been presented. The implemented computation and communication hardware enables the simultaneous execution of several computationally demanding tasks, including navigation and computer vision. Furthermore, the hardware can be easily expanded to provide more on-board computation power if required. Our data fusion enables the seamless use of different sensing paradigms with delayed information on a highly dynamic quadrotor vehicle. Its effectiveness is shown by an autonomous flight from an indoor to an outdoor environment through a 1-m-wide window, motivated by an exploration mission to enter and leave a building.

Currently, the system cannot automatically avoid obstacles. Therefore, reactive collision avoidance schemes will be implemented on the low-level system. This requires further development of object recognition and scene interpretation on resource-limited systems. As resources are limited, merely a subset of tasks can be fulfilled; therefore, we will focus on the elaboration of these tasks.

PHILIPP LUTZ

**Figure 8.** The house used for the experiments, located the DLR outdoor testbed for mobile robots. It corresponds in shape and dimensions to the house used for the IMAV exploration challenge. Provided are both an indoor environment, suitable for navigation using a laser scanner, and an outdoor environment, which is suitable for vision-based navigation.

Our future work also includes miniaturization of the system, mainly through weight reduction of the sensing equipment. For this reason, using other sensors such as

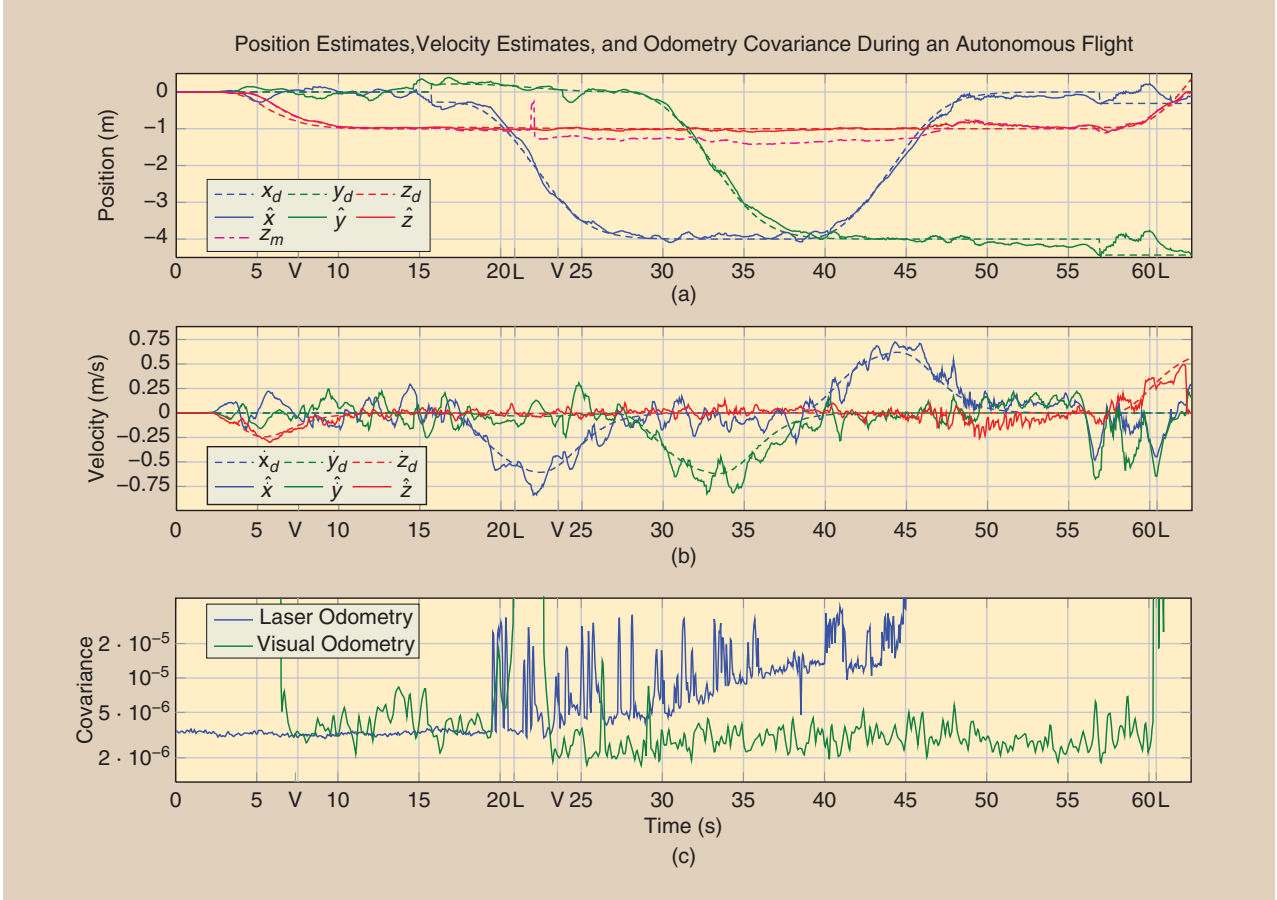



**Figure 9.** Flight from inside the house out through the window, located at $x = -2$ m. (a) The reference position $(x_d, y_d, z_d)$, estimated position $(\hat{x}, \hat{y}, \hat{z})$, and raw laser height measurement $(z_m)$. (b) The reference velocity $\dot{x}_d, \dot{y}_d, \dot{z}_d$ and estimated velocity $(\hat{\dot{x}}, \hat{\dot{y}}, \hat{\dot{z}})$. (c) The magnitude of laser and visual odometry covariances. Also shown on the time axis are indicators when the system has switched to visual ($V$) or laser ($L$) odometry. The covariance plot goes out of scope when a particular odometry is unavailable or too imprecise.

omnidirectional cameras and sonar will be investigated. Instead of stereo cameras, the Microsoft Kinect is also a viable sensor for obtaining depth images. However, because it uses artificial infrared lighting, it is only suitable for indoor applications.

In USAR missions, it might be necessary to fly to globally defined positions. This capability will be achieved by integrating GPS into the data fusion. We will also address the cooperation with multiple mobile and ground robots, as well as human interfaces to a team of robots. Currently, our fusion system is based on local navigation, yet the navigation will be improved by having higher level position information, e.g., by using a topological map. In this way, the system is enabled to navigate through large environments on a strongly hardware-limited system. Supplementary information about the DLR multicopters can be found online [20].

## References


[1] L. Lin, M. Roscheck, M. Goodrich, and B. Morse, "Supporting wilderness search and rescue with integrated intelligence: autonomy and information at the right time and the right place," in *Proc. 24th AAAI Conf. Artificial Intelligence*, Atlanta, GA, 2010, pp. 1542–1547.

[2] M.A. Goodrich, J.L. Cooper, J.A. Adams, C. Humphrey, R. Zeeman, and B.G. Buss, "Using a mini-UAV to support wilderness search and rescue: Practices for human–robot teaming," in *Proc. IEEE Int. Workshop Safety, Security and Rescue Robotics (SSRR)*, Rome, Italy, Sept. 2007, pp. 1–9.

[3] J. Casper and R.R. Murphy, "Human–robot interactions during the robot-assisted urban search and rescue response at the World Trade Center," *IEEE Trans. Syst. Man, Cybern. B: Cybernet.*, vol. 33, no. 3, pp. 367–385, 2003.

[4] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proc. IEEE ICRA*, Kobe, Japan, May 2009, pp. 2878–2883.

[5] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unstructured and unknown indoor environments," in *Proc. European Micro Aerial Vehicle Conf (EMAV)*, Delft, The Netherlands, 2009, pp. 1–8.

[6] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Proc. ICRA*, Shanghai, China, May 2011, pp. 20–25.

[7] M. Achtelik, S. Weiss, and R. Siegwart, "Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments," in *Proc. IEEE ICRA*, Shanghai, China, 2011, pp. 3056–3063.

[8] IMAV2011—International micro aerial vehicle competition. (2011). Online. Available: http://www.imav2011.org/.

[9] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. IEEE ICRA*, Anchorage, AK, 2010, pp. 21–28.

[10] N. L. Cassimatis, J. G. Trafton, M. D. Bugajska, and A. C. Schultz, "Integrating cognition, perception and action through mental simulation in robots," *Robot. Auton. Syst.*, vol. 49, no. 1–2, pp. 13–23, 2004.

[11] M. A. Goodale and G. K. Humphrey, "The objects of action and perception," *Cognition*, vol. 67, no. 1–2, pp. 181–207, 1998.

[12] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. IEEE ICRA, 2008*, pp. 19–25.

[13] H. Hirschmüller, P. R. Innocent, and J. M. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *Int. J. Comput. Vis.*, vol. 47, no. 1–3, pp. 229–246, 2002.

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

[15] H. Hirschmüller, P. R. Innocent, and J. M. Garibaldi, "Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics," in *Proc. 7th Int. Conf. Control, Automation, Robotics and Vision (ICARCV)*, Singapore, Dec. 2002, pp. 1099–1104.

[16] S. Roumeliotis and J. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proc. IEEE ICRA*, Washington, DC, 2002, vol. 2, pp. 1788–1795.

[17] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," Mitsubishi Electric Research Labs., 201, Tech. Rep., 1995.

[18] V. Utkin and J. Shi, "Integral sliding mode in systems operating under uncertainty conditions," in *Proc. 35th IEEE Decision and Control*, Kobe, Japan, 1996, vol. 4, pp. 4591–4596.

[19] K. Youcef-Toumi and O. Ito, "A time delay controller for systems with unknown dynamics," in *Proc. Am. Control Conf.*, Atlanta, GA, 1988, pp. 904–913.

[20] "German Aerospace Center (DLR) Institute of Robotics and Mechatronics." [Online]. Available: http://www.dlr.de/rm/desktopdefault.aspx/tabid-7903



***Teodor Tomić***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: teodor.tomic@dlr.de.

***Korbinian Schmid***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: korbinian.schmid@dlr.de.

***Philipp Lutz***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: philipp.lutz@dlr.de.

***Andreas Dömel***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: andreas.doemel@dlr.de.

***Michael Kassecker***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: michael.kassecker@dlr.de.

***Elmar Mair***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: elmar.mair@dlr.de.

***Iris Lynne Grixa***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: iris.grixa@dlr.de.

***Felix Ruess***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: ruessf@cs.tum.edu.

***Michael Suppa***, German Aerospace Center (DLR), Muenchner Strasse 20, 82234 Wessling, Germany. E-mail: michael.suppa@dlr.de.

***Darius Burschka***, Technische Universitaet Muenchen, Boltzmannstr. 3, 85748 Garching bei Muenchen, Germany. E-mail: burschka@cs.tum.edu.