Autonomous Corridor Flight of a UAV Using a Low-Cost and Light-Weight RGB-D Camera

Sven Lange, Niko Sünderhauf, Peer Neubert, Sebastian Drews, and Peter Protzel

Abstract. We describe the first application of the novel Kinect RGB-D sensor on a fully autonomous quadrotor UAV. In contrast to the established RGB-D devices that are both expensive and comparably heavy, the Kinect is light-weight and especially low-cost. It provides dense color and depth information and can be readily applied to a variety of tasks in the robotics domain. We apply the Kinect on a UAV in an indoor corridor scenario. The sensor extracts a 3D point cloud of the environment that is further processed on-board to identify walls, obstacles, and the position and orientation of the UAV inside the corridor. Subsequent controllers for altitude, position, velocity, and heading enable the UAV to autonomously operate in this indoor environment.

1 Introduction

One of our research projects focuses on enabling micro aerial vehicles to autonomously operate in GPS-denied environments, especially in indoor scenarios. Autonomous flight in confined spaces is a challenging task for UAVs and calls for accurate motion control as well as accurate environmental perception and modelling. RGB-D sensors are relatively new sensor systems that typically provide an RGB color image along with distance information for each image pixel and thus combine the perceptual capabilities of RGB cameras with those of stereo camera or 3D laser measurement systems.

While sensor systems like the SwissRanger or PMD cameras have been successfully used in robotics and UAV applications, they are still very expensive (>6000 €). With the very recent release of the Kinect device – an accessory to the Microsoft Xbox video game platform – very cheap (≈ 150 €) RGB-D sensors are available to

Sven Lange · Niko Sünderhauf · Peer Neubert · Sebastian Drews · Peter Protzel
Department of Electrical Engineering and Information Technology, Chemnitz University of Technology, 09111 Chemnitz, Germany
e-mail: firstname.lastname@etit.tu-chemnitz.de

the robotics community. Although precise comparisons of performance and accuracy between the established RGB-D systems and the Kinect are not yet available, it is already foreseeable that the Kinect will be a valuable sensor for a variety of robotics applications.

Our paper explores the application of the Kinect on a quadrotor UAV to aid autonomous corridor flight. After a short introduction to the sensor's working principle, we present our UAV platform and its internal system and control architecture before real-world experiments and their results are described.

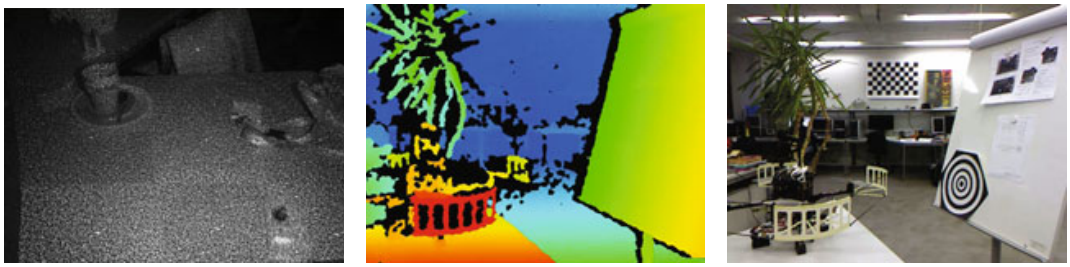
1.1 The Microsoft Kinect – A Valuable Sensor for Robotics Applications and Research

The Kinect RGB-D sensor was released by Microsoft in November 2010 as an accessory to its Xbox video game platform. Open source drivers are available from the OpenKinect project [9] or as part of the OpenNI framework [11] that can be interfaced using ROS [12].

In our work, we use the Kinect driver that is available as part of the ROS framework. This driver allows to request an RGB image, a depth image, a 3D point cloud, the raw IR image (all of resolution 640×480) and readings from the internal accelerometers of the device. Fig. 1(b) and Fig. 1(c) show a depth image along with the corresponding RGB image.

The device consists of two cameras and an infrared laser light source. The IR source projects a pattern of dark and bright spots onto the environment (see Fig. 1(a)). This pattern is received by one of the two cameras which is designed to be sensitive to IR light. Depth measurements can be obtained from the IR pattern by triangulation. According to the patent [2] held by PrimeSens Ltd., this is done by comparing the perceived IR pattern against a reference image and thereby determining the relative shift of groups of spots.

To compare the quality of the Kinect's depth measurements against other RGB-D devices such as the SwissRanger 4000 or PMD's CamCube, we determined the measurement repeatability. 3000 measurements were taken on a static planar object in distances of 2 and 4 meters. The standard deviations of these measurements were



(a) (b) (c)

Fig. 1 (a) The infrared light pattern projected by the Kinect sensor. (b) 640×480 depth image (c) RGB image of the same scene as captured by the Kinect.

Table 1 A basic comparison of three RGB-D systems. We determined the values for the repeatability of the Kinect's measurements by taking 3000 measurements in a distance of 2 and 4 meters respectively. All other values are taken from publicly available datasheets. Notice further that we reduced the weight of the Kinect to 200 g by removing the unnecessary housing.

Sensor	max. range	resolution	field of view in deg	repeatability in mm (1σ)	weight
Kinect	10 m	640×480	57.8×43.3	7.6 @ 2 m, 27.5 @ 4 m	440 g
SwissRanger 4000	8 m	176×144	43×34 / 69×56	4 / 6	470 g
PMD CamCube 3.0	7 m	200×200	40×40	3 @ 4 m	1438 g

determined to be 7.6 and 27.5 mm respectively, which is (especially at the larger distance) considerably larger than the values provided in the datasheets of the two established devices. However, considering the significantly lower price, we expect these values to be sufficient for most robotics applications. Table 1 compares the basic features of the Kinect, SwissRanger 4000 and PMD CamCube 3 RGB-D devices.

1.2 Related Work

RGB-D devices have been used by different groups of researchers for depth perception, SLAM and navigation on both ground based and aerial robots: Morris et al. [8] use a SwissRanger 4000 RGB-D camera for 3D indoor mapping planned for use on a quadrotor UAV. Henry et al. [5] describe an approach of using an RGB-D sensor to construct dense 3D models of indoor environments.

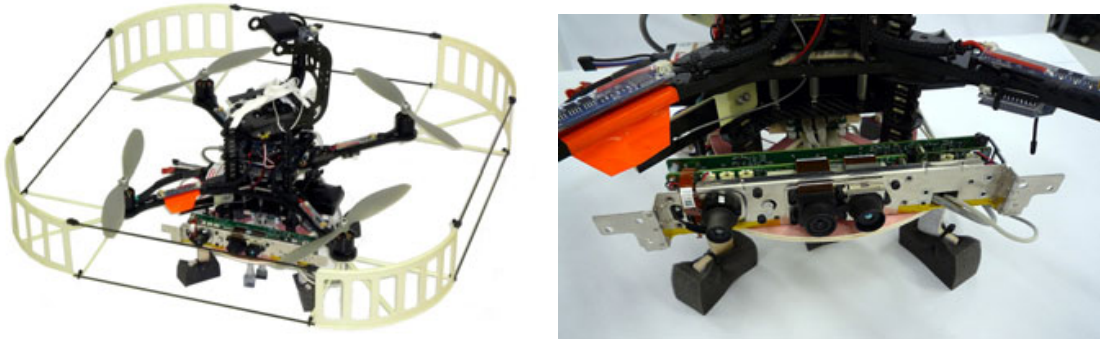
Autonomous navigation of UAVs in GPS-denied indoor environments using Hokuyo laser range finders instead of RGB-D devices has been demonstrated by [1] while [3] focused on porting SLAM algorithms that were previously developed for ground based robots to UAVs.

In previous publications [7] [6], we described our work with smaller autonomous UAVs (type "Hummingbird") whereas this paper contains updated material on the system architecture and controller structure on the new and larger "Pelican" quadrotor. To our knowledge, this paper is the first scientific publication that describes the application of the Kinect RGB-D device on an autonomous UAV.

2 Hardware and Software Architecture of Our UAV "Pelican"

The UAV we use in our project is a "Pelican" system (see Fig. 2(a)) that is manufactured by Ascending Technologies GmbH, Munich, Germany. This mid-size four-rotor UAV, or quadrocopter, measures 72 cm in diameter and can carry up to 500 g of payload for about 20 minutes.

The Pelican is propelled by four brushless DC motors and is equipped with a variety of sensors: Besides the usual accelerometers, gyros and a magnetic field sensor, a



(a) (b)

Fig. 2 (a) Our modified *Pelican* quadrotor system. (b) The dismantled Kinect sensor mounted on the UAV.

pressure sensor and a GPS module provide input for AscTec's sophisticated sensor fusion algorithm and the control loop running at 1 kHz. Like on the smaller Hummingbird system, an "AscTec AutoPilot board" is responsible for data fusion and basic control of the UAV. More technical details on the AutoPilot Board, the Hummingbird and the controllers can be found in [4]. Especially outdoors where the control loop can make use of GPS signals to enter the GPS position hold mode, the Pelican is absolutely self-stable and requires no human pilot to operate. In most cases, the deviation from the commanded hover position is below 1 m. However, when GPS signals are not available, the UAV tends to begin drifting very quickly. Therefore, when the UAV should be operated in GPS-denied environments (for instance close to buildings or indoors) a position stabilization that is independent from GPS signals is required.

With respect to the Pelican's standard configuration, our system is using additional features offered by AscTec. A large propeller protection *(135 g)* is added to minimize the effects of contact with obstacles. An embedded PC system is used for onboard computing. It is equipped with a CoreExpress 1.6 GHz Intel Atom (Z530) processor board from Lippert Embedded Computers GmbH with 1 GB of RAM including a WiFi module *(100 g)*.

2.1 Additional Custom Made Hardware and Sensors

We extended the UAV's configuration using the available payload and equipped the quadrocopter with additional hardware. This includes an SRF10 sonar sensor *(10 g)*, a microcontroller board based on an ATmega644P *(25 g)*, an ADNS-3080 optical flow sensor board based on an ATmega644P as well *(25 g)*, and a Kinect RGB-D sensor *(200 g)*. Completely equipped the quadrocopter has an overall weight of 1675 g including LiPo batteries *(380 g)*, frame and cables. All components and their communication paths are shown in Fig. 3(a).

To ensure realtime execution, the controllers are implemented on a custom made microcontroller board. It is connected to the quadrocopter via USART for polling the quadrocopter's internal sensor readings and sending flight commands. The second

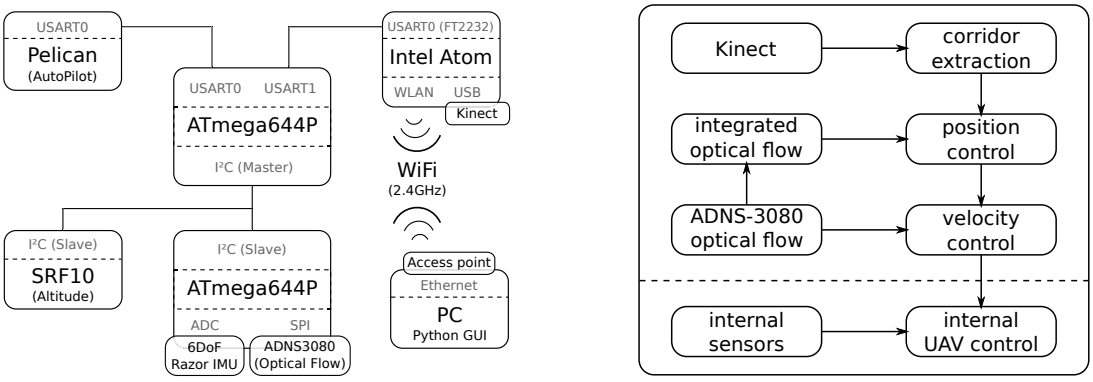


Fig. 3 (a) Schematic diagram of the main components and their communication channels. (b) Cascaded controller structure and main sensors used. For reasons of clarity the altitude controller is not shown here. It operates parallel to the position and velocity controller and uses an SRF-10 sonar sensor to measure the altitude.

USART port of the microcontroller is connected to the Atom processor board for transmitting and receiving status messages and commands from a ground station as well as from the Atom processor board itself. Communication with the ground station is realized by the integrated WiFi module.

The sensors used in the controller architecture are connected to the microcontroller via I^2C bus, acting as slave devices. An SRF10 sonar sensor measures the current altitude over ground with high precision. Furthermore the Avago ADNS-3080 optical flow sensor combined with a small and light-weight camera lens determines the UAV's current velocity over ground. The sensor is not connected directly over the I^2C bus, but is interfaced via SPI to another ATmega644P which is acting as an I^2C slave device. Additionally, the optical flow sensor board is equipped with a high power LED to enhance the optical flow performance in less illuminated environments and a separate IMU sensor for high frequency gyro measurements.

2.2 Controller Structure

In order to achieve a stable flight behavior in GPS-denied areas, we used an altitude controller and a cascaded controller structure for position stabilization. Figure 3(b) shows the principal structure along with the main sensors used by the position and velocity controllers.

2.2.1 Altitude Controller

The altitude controller receives its input from an SRF10 sonar sensor. This off-the-shelf sensor commonly used in robotics projects provides accurate altitude information and offers considerable advantages because of its short minimum operation range of 3 cm. The altitude controller itself is implemented on the ATmega644P microcontroller board as a standard PID-controller. The control loop operates on a 25 Hz cycle and is able to stabilize the UAV's altitude with an accuracy of 3 cm.

The standard PID altitude controller is very sensitive to steps in its measurements or setpoint values, which are common when flying over obstacles. If they appear, an overshoot is the common reaction of the quadrocopter. For that reason we implemented a step detection combined with a ramp function for both cases.

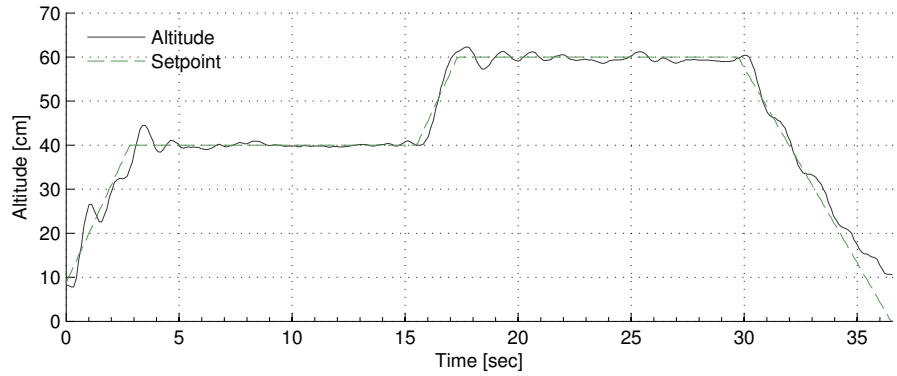


Fig. 4 Altitude plot showing a take-off, altitude-hold at 40 cm followed by an altitude-hold at 60 cm and a landing maneuver. The altitude setpoint is automatically increased and decreased in a ramp. Note that the sonar sensor is mounted about 10 cm above ground.

In Figure 4 we show the results of our altitude controller running on the Pelican UAV. The plot starts when controlled flight mode is activated and the human operator requested an altitude of 40 cm. Notice that the setpoint used by the altitude controller is increased using a ramp function to avoid overshoots caused by steps in the setpoint. After 16 seconds, the setpoint was increased to 60 cm and after an additional time of 13 seconds, landing was initiated. The implemented PID-controller is able to follow the setpoint very closely and stabilizes the UAV at the requested altitude with a maximum deviation of 3 cm.

2.2.2 Position and Velocity Controller

When no GPS signals are available, the only way to measure position and velocity with the standard platform is to integrate information from the onboard acceleration sensors and gyros. However, due to the noisy input signals large errors accumulate quickly, rendering this procedure useless for any velocity or even position control.

In our approach, an optical flow sensor facing the ground provides information on the current velocity and position of the UAV. The Avago ADNS-3080 we use is commonly found in optical mice and calculates its own movements based on optical flow information with high accuracy and a framerate of up to 6400 fps. After exchanging the optics and attaching an M12-mount lens with a focal length of 8 mm to the sensor, we are able to retrieve high quality position and velocity signals that accumulate only small errors during the flight. The sensor also provides a quality feedback which correlates with the number of texture features on the surface the sensor is facing. If the surface is not textured enough, the quality indicator drops and the velocity and shift signals become more noisy. Likewise the quality drops in cases of ill-illuminated textures, so we added a high power LED with directed light

(10° FOV) and a proper wavelenght *(623 nm)* matching the maximum responsivity of the optical flow sensor.

To calculate the velocity in metric units relative to the ground, the sensor resolution, FOV, height and changes in orientation are used. With respect to previous work [7] we increased the lens' focal length from 4.2 mm to 8 mm to obtain a higher spatial resolution and so improve velocity estimation. Because the sensor is now more sensitive to rotational changes, it is insufficient to use the quadrocopter's internal measurements for angle compensation which are available at a rate of about only 7 Hz. Therefore, we extended our optical flow sensor board by a simple 6 degrees of freedom IMU system to get gyro measurements at a much higher rate (> 50 Hz).

Both the velocity and position controller are implemented as standard PID and P controllers, respectively, and operate at a frequency of 25 Hz. In combination, they are able to stabilize the UAV's position and prevent the drift that quickly occurs when the GPS-based position hold mode is inactive.

3 Autonomous Corridor Flight Using the Kinect RGB-D Device

The Kinect driver of ROS provides a 640×480 3D point cloud that is downsampled (thinned) to approximately 3,000 points before it is processed further. This downsampling is done to increase the performance of subsequent steps in the algorithm. Details on our specialized and efficient downsampling algorithm are provided in section 3.1. After downsampling, large planar sections are found in the approximately 3,000 remaining points by applying a sample consensus (MLESAC) based parameter estimation algorithm. Fig. 5 visualizes the results. The Point Cloud Library [10] already provides convenient algorithms that extract the planes in their parameter form $ax + by + cz + d = 0$.

Given these parameters, the distance of the RGB-D device from each plane is calculated as $\Delta_i = |d_i| / \sqrt{a_i^2 + b_i^2 + c_i^2}$. Roll, pitch, and yaw angles are calculated from the plane parameters as well, e.g. the yaw angle ϕ_i is given by $\phi_i =$

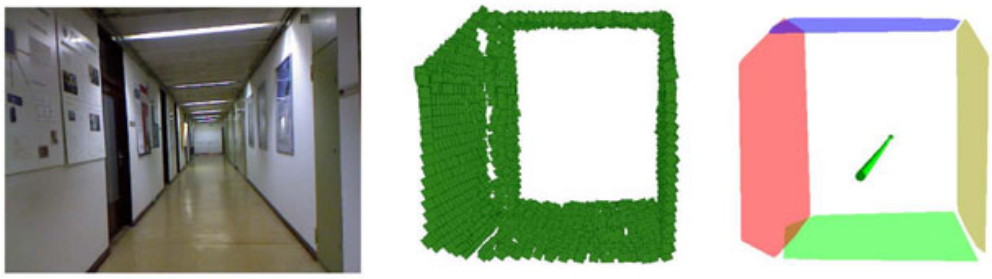


Fig. 5 *(left)* RGB image of the corridor. *(mid)* Downsampled point cloud containing about 3,000 points. *(right)* Extracted planes of the walls. The planes have been classified into floor, ceiling, left, and right wall. The green arrow shows the intended motion direction computed from the position and orientation relative to the walls.

atan2$(c_i/d_i, a_i/d_i)$. According to these angles, the extracted planes are assigned to one of the following classes: *floor, ceiling, left, right, front*.

To keep the UAV aligned with the corridor axis and in the center of the corridor, motion commands $(dx, dy, d\phi)$ are calculated from the plane distances Δ_i and the yaw estimates ϕ_i.

dx controls the forward movement along the corridor. It is always set to 1 meter, as long as no obstacle or wall is detected in front of the UAV. The desired horizontal displacement (perpendicular to the corridor walls) is given by $dy = (w_{left} \cdot (a - \Delta_{left}) + w_{right} \cdot (\Delta_{right} - a))/(w_{left} + w_{right})$ where a is half of the corridor width. w_{left} and w_{right} are weight factors and are equal to the number of scan points in the left and right plane respectively. This way, the wall that is supported by more scan points has a stronger influence on the resulting motion command. The same weight factors are used to generate the desired rotation command $d\phi$ that keeps the UAV aligned with the corridor axis: $d\phi = -\sum w_i \cdot \phi_i / \sum w_i$

The motion commands are transformed into the body coordinate system of the UAV by using $d\phi$ and then sent to the position and heading controller on the ATmega644P microcontroller board for execution. Commands are sent at a rate of approximately 4 Hz. The cascaded position and velocity controllers we described in section 2.2.2 are able to follow the commanded trajectories and thus keep the UAV in the corridor center. Although the altitude above the floor level can be estimated from the plane parameters as well, altitude control relies solely on the SRF10 sonar sensor, as a high measurement frequency is crucial for stable altitude control.

A video that shows an example flight is available at our website `www.tu-chemnitz.de/etit/proaut/forschung/quadrocopter.html.en`. Fig. 6 shows the position estimates Δ_{left} and Δ_{right} inside the corridor while performing autonomous flight. According to these internal measurements, the maximum deviation from the corridor center was 35 cm. Mean velocity was $0.36\frac{m}{s}$. At the moment, no ground truth information is available for a more detailed and quantitative analysis. We are working towards integrating an external 3D position measurement system to be able to evaluate the system's performance in depth in future work.

3.1 Point Cloud Downsampling

Point cloud downsampling is an effective way to reduce the computational effort for subsequent processing steps. To achieve homogeneous coverage in the three dimensional space, downsampling has to be done in voxel space, contrary to downsampling in the image plane of the Kinects depth image. The PCL (Point Cloud Library) [10] already provides a mechanism for voxel downsampling. Table 2 shows the time consumption of the PCL algorithm for downsampling a point cloud of 307,200 points to about 3,000 points and compares it to the time consumption of subsequent steps in the overall algorithm. It can be seen that the downsampling takes approximately 25 times as long as the wall extraction and position estimation together and thus absolutely dominates the overall runtime. To reduce the effect of this bottleneck, we implemented a specialized downsampling algorithm for our application.

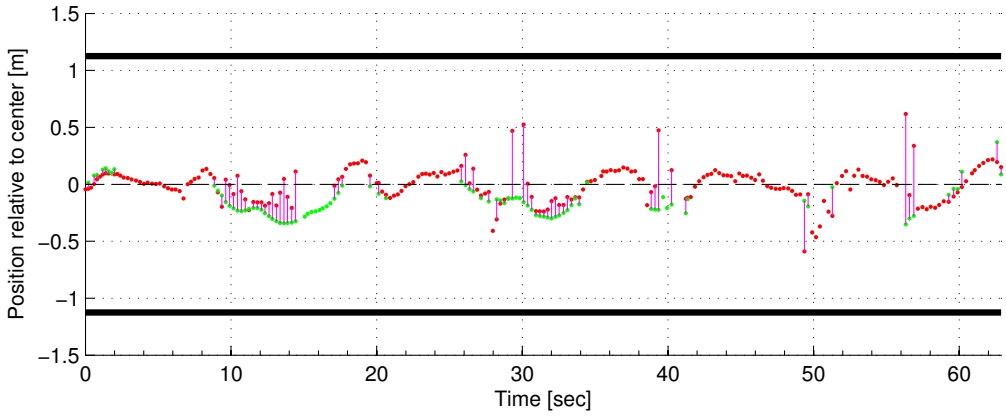


Fig. 6 Position estimates of the quadrotor within the corridor during autonomous flight, based on the Kinect measurements. The red points represent the calculated distances based on the left wall and the green points are based on the right wall. If two walls are visible, measurements are connected through a line. The corridor walls are shown as thick black horizontal lines. The maximum deviation from the corridor center was 35 cm.

While the PCL downsampling algorithm processes point clouds of various extents and data types, we use our prior knowledge about the range of the sensor, data type and data storage to speed-up the computations. Given the maximum range of the Kinect sensor and its field of view, we define a three dimensional grid of the desired resolution and size. While iterating the data of the input point cloud message, each grid cell contains just the Boolean value whether we have already processed a point inside the cell's corresponding space or not. Whenever an input point in an unoccupied grid cell is visited, it is stored in the resulting, downsampled point cloud. Further speed-up is achieved by directly processing the data in the ROS message to avoid the additional conversion from ROS message to PCL point cloud type.

By using our optimized downsampling routine, we gained an overall speed-up of factor 5.

Table 2 Time consumption comparison for point cloud downsampling methods and the other steps of the corridor extraction measured on the onboard Atom-based embedded system.

Step	Time in ms
Point cloud downsampling and conversion with PCL	492
Point cloud downsampling and conversion using our own method	89
Wall extraction, position estimation, trajectory generation	19

4 Conclusions and Future Work

Our first experiences with the novel Microsoft Kinect RGB-D device clearly showed that the sensor can be applied beneficially to mid-sized UAVs like the Pelican. We expect to see the Kinect and similar devices being used in a variety of robotics application in the near future, both in research and industry.

We demonstrated how the Kinect's depth data can be used to autonomously navigate a UAV in a corridor, without using any external sensor equipment like motion capture systems. The sourcecode for efficient point cloud downsampling and trajectory generation is available to the community as part of our ROS package at http://www.ros.org/wiki/tuc-ros-pkg. The next natural step in our work will be to extend the UAV's capabilities by enabling it to use the 3D data to avoid static and moving obstacles, and plan trajectories in less well defined indoor spaces, including large open areas as well as cluttered and confined spaces.

Acknowledgements. This work has been partially funded by the European Union with the European Social Fund (ESF) and by the state of Saxony.

References

1. Achtelik, M., Bachrach, A., He, R., Prentice, S., Roy, N.: Autonomous Navigation and Exploration of a Quadrotor Helicopter in GPS-denied Indoor Environments. In: First Symposium on Indoor Flight Issues (2009)
2. Freedman, et al: US2010/0118123A1 (2010)
3. Grzonka, S., Grisetti, G., Burgard, W.: Towards a Navigation System for Autonomous Indoor Flying. In: Proc. of IEEE International Conference on Robotics and Automation, ICRA 2009, Kobe, Japan (2009)
4. Gurdan, D., Stumpf, J., Achtelik, M., Doth, K.-M., Hirzinger, G., Rus, D.: Energy-efficient Autonomous Four-rotor Flying Robot Controlled at 1 kHz. In: Proc. of IEEE International Conference on Robotics and Automation, ICRA (2007)
5. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In: Proc. of International Symposium on Experimental Robotics, ISER 2010 (2010)
6. Lange, S., Protzel, P.: Active Stereo Vision for Autonomous Multirotor UAVs in Indoor Environments. In: Proc. of the 11th Conference Towards Autonomous Robotic Systems, TAROS 2010, Plymouth, Devon, United Kingdom (2010)
7. Lange, S., Sünderhauf, N., Protzel, P.: A Vision Based Onboard Approach for Landing and Position Control of an Autonomous Multirotor UAV in GPS-Denied Environments. In: Proc. of the International Conference on Advanced Robotics, ICAR (2009)
8. Morris, W., Dryanovski, I., Xiao, J.: 3D Indoor Mapping for Micro-UAVs Using Hybrid Range Finders and Multi-Volume Occupancy Grids. In: Proc. of Workshop on RGB-D: Advanced Reasoning with Depth Cameras in Conjunction with RSS 2010 (2010)
9. OpenKinect (2010), `http://www.openkinect.org/`
10. The Point Cloud Library (2010), `http://www.pointclouds.org/`
11. OpenNI (2011), `http://www.openni.org/`
12. ROS (2011), `http://www.ros.org/`