STEREO-BASED OBSTACLE MAPPING FROM A HELICOPTER PLATFORM

Franz Andert and *Lukas Goormann*
German Aerospace Center, Institute of Flight Systems
Lilienthalplatz 7, 38108 Braunschweig, Germany
{Franz.Andert, Lukas.Goormann}@dlr.de

OVERVIEW

This paper presents a part of the sense and avoid system of an unmanned helicopter, that describes the process of creating three-dimensional obstacle maps used by a collision avoidance system. Classical robotic approaches are combined with aircraft-specific techniques and extended with new features to meet the requirements of flight scenarios. The maps are created and updated onboard and in real-time, and it is possible to integrate a-priori knowledge and the data of other sensors or vehicles. Furthermore, the map is adaptive so that the vehicle is not restricted at all, e.g. to a pre-defined area.

1 INTRODUCTION

Sense and avoid is one of the actual challenges for unmanned systems, amongst others in the aircraft domain. The goal is to create a system that can act like a pilot – see what is around the vehicle, detect obstacles and other imminent dangers, and plan a collision-free trajectory around them. This requires sensors like radar, laser scanners or cameras for environmental perception and the hardware and software for data analysis and automatic control.

In principle, there are two ways how to implement obstacle avoidance. The first, maybe more simple way is adapted from insects and converts sensor data directly into reactive movement instructions. Successful results have already been presented for aerial vehicles [1, 2] and there are are also other researches in this direction [3].

The other way is to plan a safe flight trajectory through obstacle maps. Common approaches use e.g. force fields, vector field histograms [4] or velocity obstacles [5]. Since an autonomous vehicle must be able to operate in unknown environments, an obstacle map has to be created with the help of sensors before a path planning algorithm can be applied.

Following the literature [6, 7, 8], an empty map without any obstacle is used as initialization if the environment is completely unknown. Otherwise, already known objects are inserted. In both cases, the map is checked for changes while sensing. The advantage of this approach is that maps make it possible to save and to re-use sensor data efficiently and to plan paths in global and local environments considering the vehicle's limitations and actual map updates.

A procedure of creating obstacle maps from a UAV is described here. For sensing, a stereo camera is used. This sensor is passive and has a low weight and energy consumption which makes this approach also applicable to smaller vehicles than the used one. The vehicle and its hardware is described briefly in the following section.

2 ARTIS – A FLYING ROBOT

2.1 Overview

The rotorcraft testbed ARTIS[1] [9] has been improved over the years and a new larger helicopter called *maxiARTIS* (fig. 1) is used for the application presented here. It has a main rotor diameter of 3 meters and a total weight of up to 25 kg, including payload. Flights of more than 30 minutes are possible.

FIG. 1: The helicopter *maxiARTIS*.

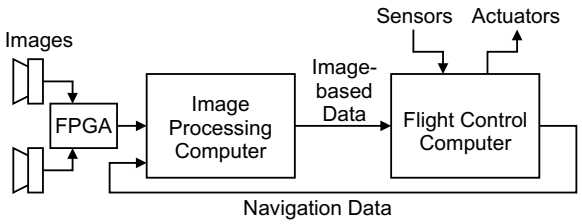

FIG. 2: Overview of the onboard hardware for vision applications.

[1] Autonomous Rotorcraft Testbed for Intelligent Systems

In addition to the vision sensor, the helicopter's position and attitude is provided in six degrees of freedom by a navigation solution using a GPS sensor, a magnetometer and an inertial measurement unit. Figure 2 illustrates the connection between vision hardware and flight controller.

2.2 Vision Hardware

For the application described in this paper, the helicopter is equipped with a stereo camera for depth measurement and a separate vision computer for camera control and image processing, see figure 3 for a detailed view. A stereo camera with a baseline of 30 cm and a field of view of approximately $51° \times 40°$ is used. It creates images with 640×480 pixels and has an inbuilt FPGA processor that calculates a depth image out of the two input images in realtime with 30 Hz. This image, see figure 4, is a result of a complex processing step where regions of the two camera images are matched [10], and it acts as a depth sensor in the mapping process.

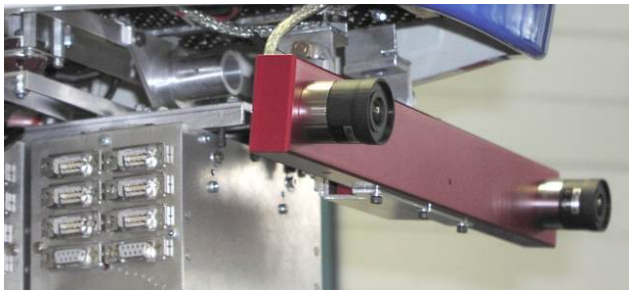

FIG. 3: The vision computer and the stereo camera.

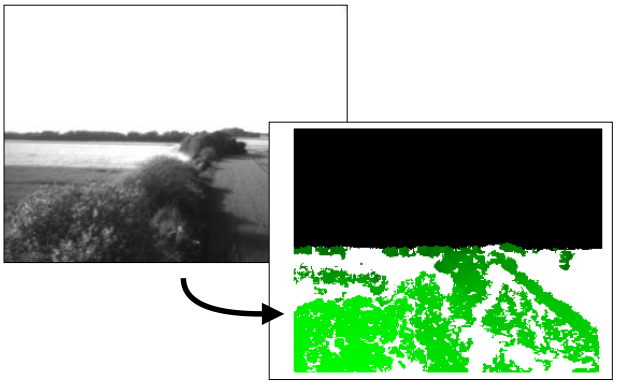

FIG. 4: Left camera image and generated depth image of this stereo pair. Darker green values represent greater distances, missing values are marked white. The sky has been detected separately and is marked black, meaning infinite distance.

2.3 Navigation Solution

The helicopter's position and attitude is provided by the flight control computer using a differential GPS sensor, a magnetometer and an inertial measurement unit. The raw data of these sensors are integrated by an Extended Kalman filter [11] to provide an accurate solution in all six degrees of freedom. Table 1 shows the accuracy of this filter calculated out of typical output data sequences. The exact deviation values may vary due to variable input accuracy, especially of the GPS sensor.

latitude	0.20 m
longitude	0.20 m
height	0.50 m
yaw	3.0°
pitch	4.0°
roll	0.7°

TAB. 1: Typical standard deviations of the navigation data.

Filtered navigation data is sent with a rate of 100 Hz to the vision computer. All computer clocks are synchronized so that a fitting recording pose of an image with a given timestamp can be obtained.

3 ROBOTIC MAPPING

3.1 Occupancy Grids

The basic mapping method follows classical approaches with occupancy grids [6, 12]. These grids have turned out to be very useful for obstacle mapping since they allow easy sensor fusion, reduce sensor noise and are also applicable to multiple vehicles. Here, a world-centric 3-D grid represents the map. Each cell consists of a value describing the presence of obstacles. Higher values refer to a higher probability of being occupied. The map is created incrementally by starting with an empty grid and writing the actual sensor information with each new depth image.

A single depth image is interpreted as follows: Each pixel refers to an object coordinate in the world and to free space along a ray between the camera center and the object coordinate. If the distance encoded by a pixel exceeds a threshold, this image point leads to free space at all. A line is drawn for each pixel, illustrated by figure 5. The second case considers uncertainty of depth measurement. With that, the viewable area is spanned and the obstacles are drawn into.

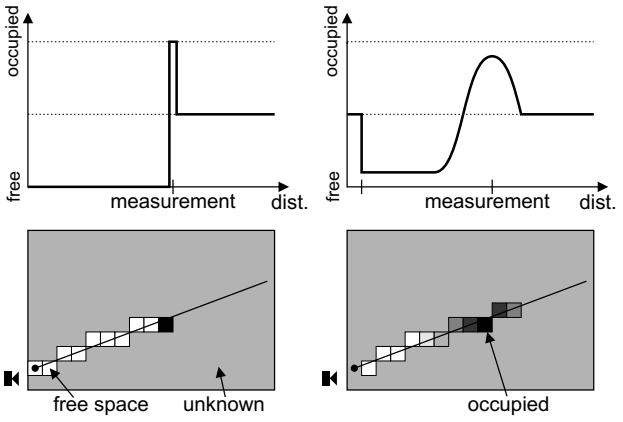

FIG. 5: Interpretation of an ideal depth image pixel (left) and a more realistic model (right).

The result of processing all rays of a single image are illustrated in figure 6. Obstacles (red) and free areas (blue) of the image are also visible in the corresponding map.

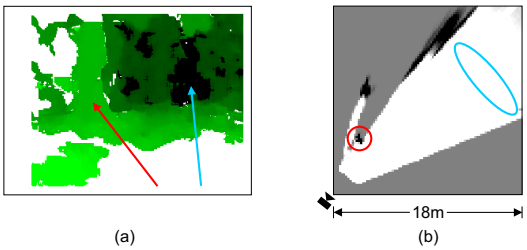

FIG. 6: Depth image example (a) and a 2-D view of the local map that was built out of it (b).

The data fusion between succeeding image frames is done by adding the map cell values. Similar to other approaches, free or occupied cells become more significant if measured several times and noise is filtered. With that, especially unmoved objects can be determined easily. In practice, the map values are truncated to a specified range so that integers can be used for the cell array. The range must be large enough to ensure the robustness to failures.

3.2 Localization

To put sensor data into a world-centric map, it is necessary to know from where the data was recorded. Acquiring and estimating the exact pose of the vehicle is a complex problem in robotics. The quality of navigation data can be improved while mapping, for example by finding the position at which the actual sensor data matches best with the map constructed so far. A lot of such Simultaneous Localization and Mapping (SLAM) approaches are presented in the literature [7, 8, 13, 14] and help to minimize drift errors and produce a consistent map.

Since obstacles are needed for orientation, the results of SLAM approaches are very ambiguous if no or just too few objects are located in the nearer environment. In cases of flights over empty fields or in areas with just a single obstacle, SLAM would not work. Nevertheless, it helps in more urban environments, especially to compensate a loss of the GPS connection.

In this case, only the navigation data is used for localization. The position accuracy of ARTIS is sufficient for maps to be used by obstacle avoidance and there is no drift due to the absolute GPS positioning. Additionally, the resulting maps are georeferenced.

As already mentioned, the pose information is known by the image processing computer. If an image has been recorded at time t, the nearest pose information $x_{t+\Delta t}$ at time $t + \Delta t$ with smallest absolute Δt is used and recalculated to a pose x_t that corresponds with the image time. The time difference is due to the different update rates and the lack of a trigger e.g. for image recording.

This calculation is done using simple movement equations and the deviations of the position that are also given by the navigation filter. Because constant accelerations and turn rates are assumed, this is an approximation that works for small time delays Δt. It is

$$x_t = x_{t+\Delta t} + \dot{x}_{t+\Delta t} \cdot (-\Delta t) + \frac{1}{2}\ddot{x}_{t+\Delta t} \cdot (\Delta t)^2 \quad (1)$$

for position vector components and

$$\phi_t = \phi_{t+\Delta t} + \dot{\phi}_{t+\Delta t} \cdot (-\Delta t) \quad (2)$$

for attitude angles. The second deviation is not used in the latter case because it is very inaccurate.

3.3 Extracting Obstacle Features

A disadvantage of grid maps is the high memory usage and that single objects are not marked separately for obstacle avoidance algorithms. Hence, it is a straightforward procedure to generate a grid map from sensor data to benefit from its advantages, and then extract features out of it for later purposes.

Features are detected by segmenting the grid into occupied and free areas applying a threshold. A single object is a set of occupied cells that are connected in a 6-neighborhood of the 3-D array. These objects are recognized with a flood fill algorithm. By saving the minimal and maximal values of the coordinates of cells belonging to the object, the bounding box is calculated and put into the global map as it is illustrated in figure 7. Unlike the cell array, these boxes need much less memory and it is easy to make them available to further applications, independent of the existence of a grid.

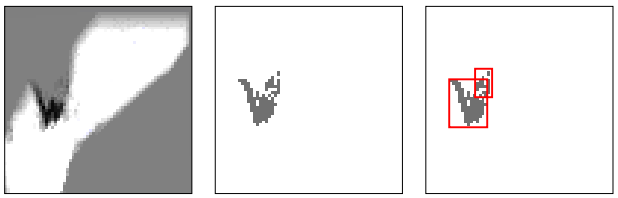

FIG. 7: Extracting features from an occupancy grid (a) with thresholding (b) and bounding box calculation (c). Bounding boxes can overlap.

Saving the bounding boxes leads to a great data reduction since they do not contain the object shapes. But they will only be a useful object representation in rather unoccupied environments, e.g. outdoor scenarios with large free spaces and single obstacles like trees. Free paths inside bounding boxes of tunnel or canyon walls will not be found in the feature map since the whole box is regarded as occupied in the feature map.

As an improvement, shape detection algorithms are applied, and the shape of each object is stored together with the box. It is an easy way to use octtrees of the cell cluster describing an obstacle. With that, the level of detail that shall be stored or sent to other applications can be adjusted easily. Additional data reduction can be achieved when assuming vertical walls. Objects are now just prisms and can be modeled as quadtrees in addition to the minimal and maximal height given by the bounding box.

3.4 Dealing with Large Environments

Since grids are usually stored as continuous data blocks in the computer memory, the boundaries of the map must be known before. If the vehicle moves outside, extensive reallocation or shifting methods must be applied because the map boundaries change.

To avoid this, the map is partitioned into fixed zones. The grid array of each zone is stored in a separate data structure so that map parts can be added or removed if its boundaries change. Additionally, the fact is used that the data fusion of the map with one image will only affect to grid cells inside a small environment of the actual position. There is no need to update cell values outside this environment. Hence, it is satisfactory to have only an occupancy grid representation of the map in zones that are visible and inside the depth sensor range.

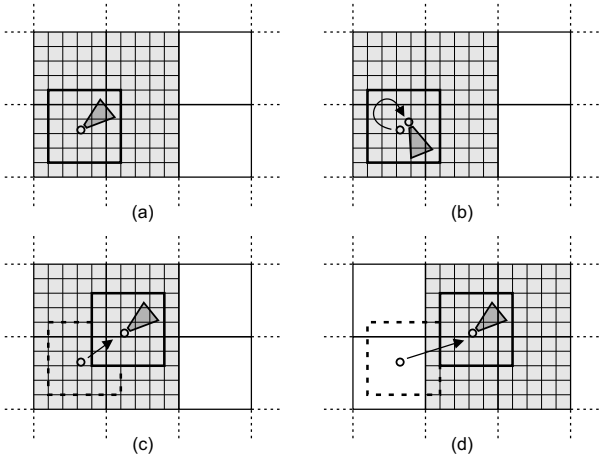

FIG. 8: 2-D view of the global map that is divided into zones. For zones around the actual camera position, an occupancy grid representation exists.

Figure 8 illustrates how the zone partition works. The actual camera and environment position is shown in subfigure (a), the other graphics show possible effects on the map, caused by the next measurement. Often, rotations (b) or movements (c) will not have an effect on the zone boundaries. But if the movement is larger so that boundaries are crossed (d), new memory is allocated for these zones. Grid information is discarded for zones that fall outside. For a fast check, just a bounding box of the helicopter and its sensor range is used.

Unlike the occupancy grid, the feature map is global. Objects are stored independently to the presence of a grid. If some grid data is removed, the corresponding features will remain (fig. 9). Vice versa, objects can be inserted into the grid (fig. 10). It is possible to include a-priori knowledge about obstacles here.

4 LABORATORY SIMULATION

4.1 Experimental Setup

As already presented in previous work [3], the flight and vision behavior is simulated before the real outdoor tests

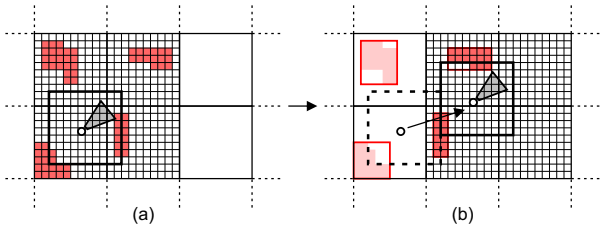

FIG. 9: Features are stored when the grid data is discarded due to helicopter movement.

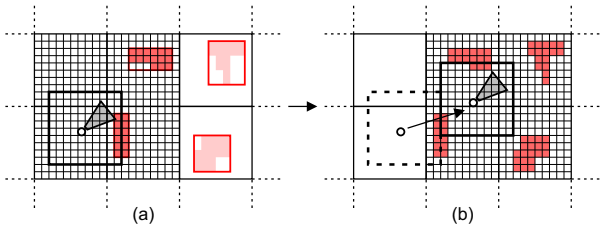

FIG. 10: Inserting features to the grid if there are objects inside a new zone.

start. The image processing part has its own “hardware-in-the-loop” simulation where the images are processed in real but the visible scene is generated and the helicopter flight done by a simulation environment that reproduces the behavior of the vehicle.

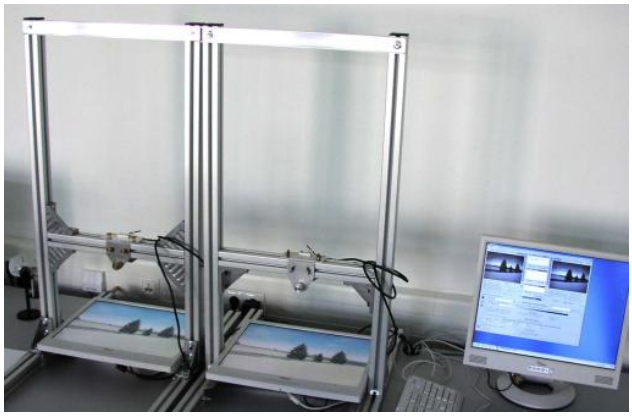

FIG. 11: Hardware-in-the-loop simulation by capturing images from displays showing generated views.

In this simulation, a generated scene is projected on two screens and captured by the stereo camera as seen in figure 11. The cameras are connected to the vision computer. The original flight control computer is also used, only the sensors are emulated and the visible scene is generated from a 3-D model of the test area. This configuration allows not only the test of the algorithms, but also a stability check of the network and the interfaces to the cameras and other sensors.

4.2 Results

A flight in a field with a line of trees is simulated as shown in figure 12. The path is a kind of slalom through

the trees that shall be mapped. The resolution of the grid is 1 meter which allows a real-time processing.

FIG. 12: Image from the simulation, recorded with the camera of the experimental setup.

For viewing purposes, a height profile representation is used. It shows a grid in x- (north) and y-direction (east). For each 2-D cell of that profile, the occupied 3-D map cell with maximal height is drawn with a height-specific color. However, holes are discarded in this representation but in the cases shown here, the information loss is negligible. Figure 13 shows a typical output.

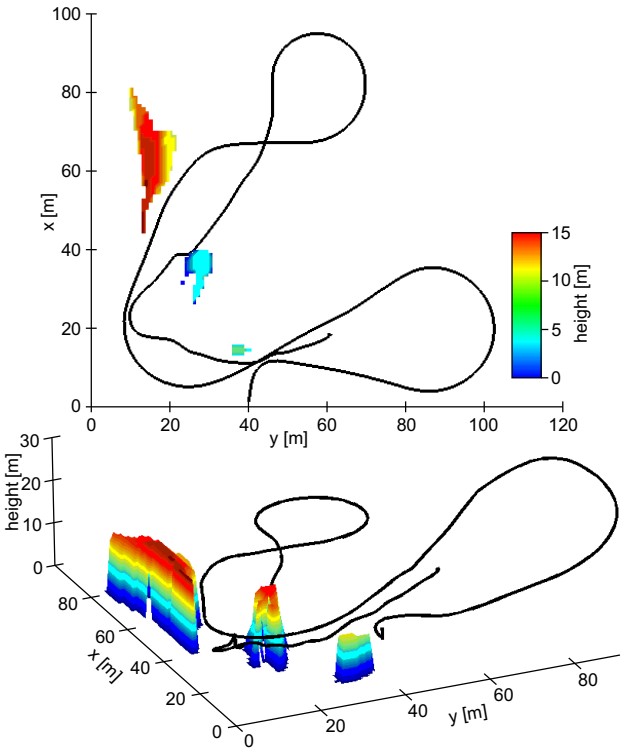

FIG. 13: Height profile map and flight trajectory of a simulation in a 2-D (top) and 3-D view (bottom).

The flight trajectory passed three trees and all of them can be distinguished from free space. The position is also correct, the objects' centers of mass lie correctly on the same line on the map. But the size especially of the top left one is too large since multiple trees have been merged to one cluster of occupied map cells. The floor has not been detected at all.

These flaws are caused by a poor depth image quality and it is assumed that these problems are mainly due to the characteristics of recording images from screens. First, there is typical aliasing in the generated views. It is possible that small object details are projected to one screen and not or modified to the other one. Second, there are moiré effects caused by the interference of the screen and camera sensor grids, visible through lines in the recorded image. And in addition, the refresh rate of the camera is slower than the screen so that two or more monitor images may be exposed at once. This is a problem especially when flying fast or when rotating rapidly.

5 OUTDOOR FLIGHT RESULTS

The goal of the helicopter flight (fig. 14) is to detect the obstacles as seen in figure 15: A straight line of bushes between the fields, and a small aircraft on the left side. A short flight between these obstacles is done here.

FIG. 14: Preparing a helicopter flight. In the background, there is a chessboard pattern for camera calibration and the ground control station truck.

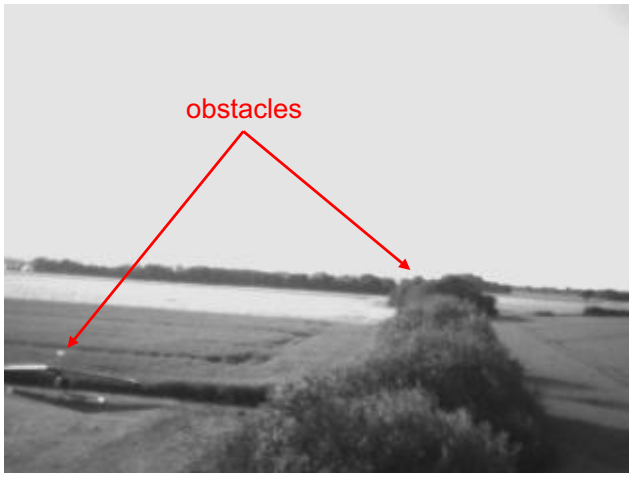

FIG. 15: Left stereo camera image, recorded during a flight.

The mapping specifications are the same as in the simulation, i.e. a resolution of 1 meter. The accuracy of posi-

tioning (see table 1) allows also a higher resolution, but the mapping procedure is slower in this case.

In contrast to the simulation, a real scene is recorded and the depth image data quality is as expected to be better even though the helicopter vibrates. The bushes and the plane have been detected correctly, and the floor too (fig. 16). However, some missing data remains due to noise. When looking at the upper end of the flight path, objects with a distance of 40 meters can be detected correctly. This method of mapping seems to be qualified for a later obstacle avoidance application.

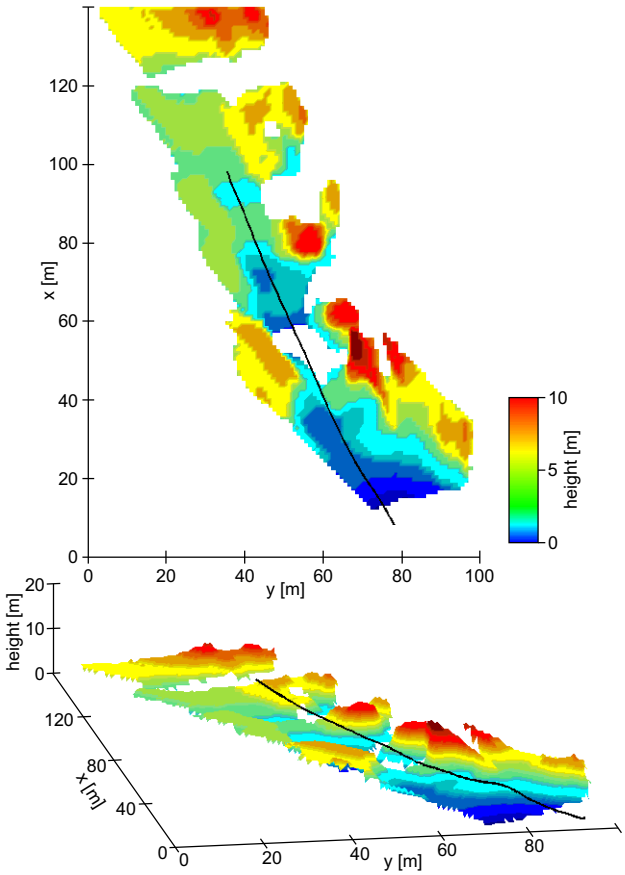

FIG. 16: Height profile map and flight trajectory of the outdoor flight in a 2-D (top) and 3-D view (bottom).

6 CONCLUSIONS

The paper has shown a technique for obstacle mapping. With the help of an occupancy grid, free areas and fixed obstacles can be detected in unknown scenarios. The grid map is available in only a small environment around the helicopter and moves along with the helicopter movement. Connected occupied cells are regarded as single objects. Their bounding box and a kind of their shape is put into a feature-based map. In contrast to the occupancy grid, the feature map is global. The map calculation is fast enough to be an input for further real-time applications.

Future work will concentrate on improvements to the global map. The recognition of polygonal shapes is one task. Another challenge is the tracking of features to handle moving objects. Applications like obstacle avoidance or autonomous path planning are currently developed.

REFERENCES

[1] S. Griffiths, J. Saunders, and A. Curtis, "Obstacle and terrain avoidance for miniature aerial vehicles," *Robotics and Automation Magazine (Preprint)*, 2006.

[2] S. E. Hrabar, P. I. Corke, G. S. Sukhatme, K. Usher, and J. M. Roberts, "Combined optic-flow and stereo-based navigation of urban canyons for a uav," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 302–309.

[3] F. Andert, G. Strickert, and F. Thielecke, "Depth image processing for obstacle avoidance of an autonomous vtol uav," in *DGLR German Aerospace Congress*, 2006, pp. 1327–1334.

[4] J. Borenstein and Y. Koren, "The vector field histogram –fast obstacle avoidance for mobile robots," *IEEE Journal of Robotics and Automation*, vol. 7, no. 3, pp. 278–288, 1991.

[5] P. Fiorini and Z. Shiller, "Robot motion planning in dynamic environments using velocity obstacles," *International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[6] H. P. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *IEEE International Conference on Robotics and Automation*, vol. 2, March 1985, pp. 116–121.

[7] D. Hähnel, "Mapping with mobile robots," Ph.D. dissertation, Universität Freiburg, 2004.

[8] S. Thrun, "Robotic mapping: A survey," Carnegie Mellon University, Tech. Rep. CMU-CS-02-111, February 2002.

[9] J. Dittrich, A. Bernatz, and F. Thielecke, "Intelligent systems research using a small autonomous rotorcraft testbed." in *2nd AIAA Unmanned Unlimited–Systems, Technologies and Operations–Aerospace*, 2003.

[10] D. Scharstein and R. Szeliski, "A taxonomy of dense two-frame stereo correspondence algorithms," Microsoft Research, Tech. Rep. MSR-TR-2001-81, November 2001.

[11] A. Koch, H. Wittich, and F. Thielecke, "A vision-based navigation algorithm for a vtol uav," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2006.

[12] U. Raschke and J. Borenstein, "A comparison of grid-type map-building techniques by index of performance," in *IEEE International Conference on Robotics and Automation*, vol. 3, 1990, pp. 1828–1832.

[13] K. Konolige, "Improved occupancy grids for map building," *Autonomous Robots*, vol. 4, pp. 351–367, 1997.

[14] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping," in *IEEE International Conference on Robotics and Automation*, 2000.