Omnidirectional Visual Obstacle Detection using Embedded FPGA

Pascal Gohl[1], Dominik Honegger[2], Sammy Omari[1], Markus Achtelik[1], Marc Pollefeys[2] and Roland Siegwart[1]

***Abstract*— For autonomous navigation of Micro Aerial Vehicles (MAVs) in cluttered environments, it is essential to detect potential obstacles not only in the direction of flight but in their entire local environment. While there exist systems that do vision based obstacle detection, most of them are limited to a single perception direction. Extending these systems to a multi-directional sensing approach would exhaust the payload limit in terms of weight and computational power.**

We present a novel light-weight sensor setup comprising of four stereo heads and an inertial measurement unit (IMU) to perform FPGA-based dense reconstruction for obstacle detection in all directions. As the data-rate scales up with the number of cameras we use an FPGA to perform streaming based tasks in real-time and show a light-weight polar-coordinate map to allow a companion computer to fully process the data of all the cameras and perform obstacle detection in real-time. The system is able to process up to 80 frames per second (fps) freely distributed on the four stereo heads while maintaining a low power budget. The perception system including FPGA, image sensors and stereo mounts is 235 g in weight.

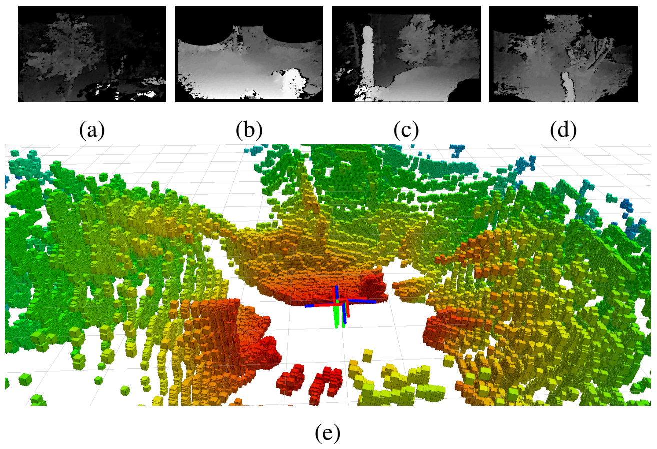

Fig. 1: Point cloud of the spherical map representation (e) with the according four disparity maps (a-d). In the front (b) and back (d) disparity map the mask for the rotors is visible in the top corners.

I. Introduction

Micro Aerial Vehicles (MAVs) are well suited for a variety of robotics applications. Due to their mobility they can be used in situations such as disaster scene surveillance, inspection and delivery of goods. For MAVs to function in cluttered environments (semi-) autonomously, it is essential that they are able to detect and avoid obstacles not only in the direction of flight but all around the vehicle.

With an omnidirectional obstacle sensing capability the MAV is more flexible in path planning as flight direction changes can be applied without rotating the MAV to face the main sensors again in the direction of flight. Also dynamic obstacles can be addressed faster and independent of the MAV's orientation. For example, navigating through a moving crowd of people or flying in a highly cluttered environment with occlusions is much safer having a sensing capability all around.

As discussed in detail in the following section, there exist many sophisticated systems showing vision-based obstacle avoidance. However most of them use a single direction sensing approach where the main image sensors are facing in the direction of flight. Additionally, these algorithms require high-power multicore CPU systems to run them at a reasonable update rate. Another option are ground station CPU systems to perform fast off-board processing but they rely on a high-latency communication link.

There is a range of other interesting depth sensing options that can potentially be integrated on small-scale MAVs for omnidirectional obstacle detection. One of the first sensors used for obstacle perception in robotics in general is the ultrasound distance sensor. The downside is the low bandwidth of the sensors due to the limitation in speed of sound as well as the relatively small detection distance and the fact that only small regions in the measurement cone can be sampled for obstacles. 2D laser range scanners can be used to detect obstacles in a two-dimensional cross-section around the MAV. The limitations of the system are its relatively large weight and the fact that laser-scanners, which are suitable for small-scale MAVs, are usually restricted to planar, 2D measurements. RGBD cameras are an interesting option for obstacle perception. However, as they are usually infrared-based they are mostly limited to indoor applications without direct sunlight.

We believe that passive, camera-based systems are ideal for MAVs as they capture rich information about the environment at a high update rate and high resolution while maintaining small size and low weight. However, this vast information of raw image data requires significant CPU resources to be processed into usable information for obstacle detection. So far, most approaches rely on CPU-based stereo densification which scales poorly with the number of used stereo camera pairs or the image resolution.

[1]Autonomous Systems Lab, ETH Zürich, Switzerland {fistname.lastname}@mavt.ethz.ch

[2]Computer Vision and Geometry Group, ETH Zürich, Switzerland {fistname.lastname}@inf.ethz.ch

Dominik Honegger is partially supported by the Swiss National Science Foundation (SNF) under grant 150151.

Pascal Gohl and Sammy Omari are supported by the Commission for Technology and Innovation (CTI) of Switzerland under grant 16432.1.

This paper shows a novel solution to address these problems: using a multi stereo camera setup in conjunction with an FPGA we are able to use simple and fast algorithms to map and detect obstacles all around the MAV. The FPGA is used to process the incoming stereo images into disparity images. More precisely, the undistortion, rectification and a dense matching is directly performed on the FPGA with a framerate of up to 80 Hz, thus significantly reducing the CPU load for perception tasks. The weight of the full sensor system with FPGA and aluminum mounts is 235 g, thus rendering it an ideal option for the use on a constrained MAV platform in terms of payload, computational power and energy consumption.

The structure of this publication is as follows: we first show related work in the field of vision based obstacle detection. Second, we provide a detailed description of the obstacle detection and mapping algorithm. We then present our test system setup with the major hardware components, followed by an efficient implementation of the obstacle detection algorithm. Lastly, we give results of the obstacle mapping and detection in a cluttered outdoor scenario.

II. Related Work

In the last decade, there has been a substantial amount of work done in enabling vision based obstacle detection on MAVs. Most approaches are based on a single direction sensing approach. High level path planning in 3D maps from a stereo head is shown in [1] and [2]. There are also approaches using an active depth sensor instead of a passive stereo head [3][4]. They all successfully show autonomous flights in unknown environments but are restricted to static scenes as they need to turn the vehicle around to sense the complete surroundings. Our system performs obstacle detection simultaneously in all directions and independent of the orientation.
A low-latency obstacle segmentation directly from depth maps is described in [5]. They use high frame rate disparity maps for obstacle detection but are again restricted to a single direction. An approach that avoids to fully process high bandwidth disparity data is shown in [6]. The push broom stereo algorithm relies on predictable motion of the vehicle towards the scene. Depth sensing is therefore limited to the direction of motion only.
In [7], the authors present a navigation scheme for operating MAVs in cluttered industrial environments using a front-looking stereo camera. [8] presents an omnidirectional sensing approach towards multimodal obstacle avoidance. Their MAV perception system is based on a 3D laser scanner, ultra sonic distance sensors and two stereo camera heads. The laser scanner provides an almost omnidirectional view of the surroundings but with an update rate of only 2 Hz. Also the laser scanner alone is already 210 g in weight that uses the payload to capacity whereas our system relies on light-weight image sensors only.
An omnidirectional sensing approach using a single camera system is shown in [9]. They use a dedicated camera and several full size computers to generate panorama disparity maps at 5 Hz update rate.
A comparable system to ours based on Intel RealSense modules on an AscTec Firefly multi-copter drones was demonstrated at CES15[1]. They performed obstacle avoidance in a cluttered environment. However they use active depth sensors, compared to our system that is based on passive image sensors only.

Statistical voxel-map approaches, such as Octomap[10], are a commonly used dense 3D map representation. The Octomap framework is strongly optimized for fast occupancy lookups, combined with a statistical outlier rejection scheme. However, inserting data is computationally expensive and the outlier rejection introduces a delay. As there is no need for a global map in the context of local robot navigation, there is a clear need for a robust and fast local map representation. In [11], the authors present a path planning and local collision avoidance scheme which operates directly on the disparity data. Equivalently, transforming disparity data from one view point into the current camera position was shown in [12] and used for disparity data filtering in [13].

Using a local polar map for collision avoidance was previously shown in [14] and in [15] combined with ultrasonic sensors. In [1] is shown how dense data can be reduced by filtering them in spherical coordinates and doing the collision avoidance in an octomap later.

III. Algorithm

In the following section we describe the algorithms performing disparity estimation, omnidirectional mapping and obstacle detection based on the data streams of the image sensors.

A. Dense Stereo

Stereo matching is performed based on real-time Semi Global Matching (SGM) as shown in [16]. The hardware synchronized image data streams of a single stereo head are stored in a buffer. Lens distortion correction and rectification are combined into a single warp operation. Radial and tangential distortion parameters as described in [17] are used to find the corrected pixel location in the buffer. Stereo matching with epipolar geometry is performed after the undistortion and rectification block. The best match based on a local cost function and global consistency constraints is selected as the valid disparity output. The corrected image data streams of both sensors are finally synchronized with the disparity output stream.

B. State-estimation

To perform the mapping, the camera rig is required to be localized in a consistent way at least locally. As we maintain a *local* map only, no globally consistent localization is required. For localization, we use the data from a single stereo pair together with the inertial data in a tightly coupled visual-inertial state estimator [18]. In order to ensure fast insertion of the disparity maps into the local map, we do not wait for the optimization procedure of the state-estimator to

[1]www.cesweb.org

finish. Instead, we simply forward-propagate the last known pose using the IMU measurements and the current IMU bias estimate.

C. Local Map

Having a map representation as close as possible to the sensing technology used for map generation allows fast insertion with the benefits of representing the complete surrounding of the MAV. In the context of an omnidirectional depth sensor, spherical depth representation can capture the local geometry of the robot's environment efficiently. Alike disparity data, spherical coordinates allow to store depth data by bearing direction and corresponding depth. Such a 2D distance map covering the sphere around the MAV can be used efficiently for local collision avoidance. We use θ to represent the horizontal angle and ϕ for the vertical angle. ρ defines the depth of a sperical map point.

The surface of the sphere is discretized into N rows, which correspond to $\theta \in [0, 2\pi]$ and M columns for $\phi \in [\phi', \pi - \phi']$. Where ϕ' is used to define a small dead zone in the polar regions, to avoid the uninteresting but very dense region at the poles.

1) Disparity handling: For outlier rejection we apply the speckle filter from OpenCV [19] directly on the disparity data. This allows us to reject small blobs of wrong matched environment, without introducing a delay longer than the computation time required to run the algorithm. By employing such a speckle filter, there is the risk that correctly detected small objects can be removed. However, in the context of the experiments in a forest setting (Sec. V-C), it is shown that the scheme is still able to detect leaves and small branches. It also provides enough outlier filtering to get a clean free space around the MAV.

To insert disparity data into the spherical map, the disparity is projected into 3D points, then transformed from the camera frame into the map frame using the transformation $\mathbf{T_{MC}}$ and finally projected into the spherical coordinate system. The disparity data provided by the FPGA SGM core are already rectified and undistorted, why we do not need to take the lens model into account at this stage.

We can directly map each pixel coordinate $\mathbf{x} = (u, v)$ and its disparity $\mathbf{D}(\mathbf{x})$ into the corresponding cartesian 3D point $\mathbf{p_C} \in \mathbb{R}^3$ represented in the camera frame

$$\mathbf{p_C} = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \frac{f \cdot b}{\mathbf{D}(\mathbf{x})} \begin{bmatrix} \frac{u}{f} \\ \frac{v}{f} \\ 1 \end{bmatrix}. \tag{1}$$

Where f is the focal length of the camera and b the length of the baseline. The transformation $\mathbf{T_{MC}}$ to transform the 3D points from the camera frame into the map frame is

$$\mathbf{T_{MC}} = \mathbf{T_{MW}}^{-1} \cdot \mathbf{T_{WS}} \cdot \mathbf{T_{SC}}. \tag{2}$$

$\mathbf{T_{SC}}$ is the static transformation of the camera frame into the MAV state frame. From the visual inertial state estimation framework used for the localization of the MAV, we get the full pose of the MAV in world frame, which provides us the transform from the MAV state to world $\mathbf{T_{WS}}$. $\mathbf{T_{MW}}$ is the transformation from the world frame to the current map frame. By applying $\mathbf{T_{MC}}$ to $\mathbf{p_C}$ we get the 3D coordinates in the map frame,

$$\mathbf{p_M} = \begin{bmatrix} x_M \\ y_M \\ z_M \end{bmatrix} = \mathbf{T_{MC}} \cdot \mathbf{p_C}, \tag{3}$$

which than can be converted to spherical coordinates:

$$\begin{aligned} \rho &= \sqrt{x_M^2 + y_M^2 + z_M^2} \\ \theta &= \text{atan2}\left(y_M, x_M\right) \\ \phi &= \arccos\left(\frac{z_M}{\rho}\right). \end{aligned} \tag{4}$$

2) 3D data lookup: For real time obstacle detection and collision avoidance, fast data access of map data is essential. The fastest and simplest access is looking up a distance measurement by giving a bearing angle. Bearing angles can be mapped directly to the discrete map storage indexes u and v by

$$u = \frac{\theta + \pi}{r_u} \text{ and } v = \frac{\phi - \phi'}{r_v}, \tag{5}$$

where r_u is the horizontal resolution of the map and r_v the vertical resolution. Using the map indexes, the distance can directly be read out of the memory

$$\rho = \mathbf{M}(u, v). \tag{6}$$

For a more sophisticated motion planning system, a point cloud of close obstacles can be easily extracted by iterating over a range of bearing angles,

$$\mathbf{p_M} = \begin{bmatrix} \rho \cdot \sin(\phi) \cdot \cos(\theta) \\ \rho \cdot \sin(\phi) \cdot \sin(\theta) \\ \rho \cdot \cos(\theta) \end{bmatrix} \tag{7}$$

where ρ is from Eq. 6 with u and v from Eq. 5.

3) Map Moving: To keep the map local, the depth data in the map have to be transformed to the current position while moving the MAV. This warping is computationally expensive and introduces discretization errors due to the low angular resolution of the map storage. To reduce the influence of these errors, the map frame is only updated after a certain minimal translation of the MAV. This introduces a small parallax effect when inserting new data, due to a new view point of the camera in respect to the map center point. Especially when looking at a edge of an obstacle, regions behind can be occluded from the viewpoint of the map center, but be visible at the new camera position. When taking only the closes depth while inserting data into one map point multiple times, the parallax errors can be handled in a way such that no safety relevant information is lost. After the map is moved the void in previously occluded regions will be filled as soon as new depth data are processed.

When the norm of the MAV's translational motion exceeds a certain threshold, the map center is transformed into the current MAV position using the transform from the map frame to the current MAV state.

$$\mathbf{T_{MS}} = \mathbf{T_{WS}}^{-1} \cdot \mathbf{T_{WM}}. \tag{8}$$

Each data point in the old map is converted to Cartesian coordinates using Eq. 7 and then transformed to the new map frame

$$\mathbf{p}_{new} = \mathbf{T}_{\mathbf{MB}} * \mathbf{p}_{old} \quad (9)$$

followed by inserting it into the new map using Eq. 4.

D. Obstacle Detection

For a potential simple obstacle avoidance system, we extract all the points in the map, which are closer than a certain threshold. This list can be generated very efficiently by iterating over all the map indexes. In this step, the low resolution (typically 1000×500) of the map is the major speedup factor compared to working directly on the four disparity maps. The 2D property of the local map enables fast iteration over all map points, and therefore over all directions around the MAV. This is also in contrast to the voxel-grid Octomap-based 3D representation which requires a large set of ray casts to properly sample the local environment of the MAV.

IV. System Setup

This section shows the general system structure of the MAV and the multi camera setup including an overview of the involved processing software and hardware for obstacle detection.

We use eight image sensors, configured as four stereo heads, to perform disparity estimation in all directions. All image sensors are connected to a single FPGA. An intellectual property (IP) block interfaces the image sensors and maintains pairwise synchronization of the sensors of the individual stereo heads. A subsequent distortion correction and rectification block aligns the image streams of a stereo head to enable epipolar geometry. The IP blocks performing computer vision tasks are time shared among all stereo heads. The blocks automatically detect which head is streaming data and load the corresponding calibration files. Stereo matching based on an SGM algorithm is performed afterwards. The corrected images next to disparity maps are sent to the companion computer via an Ethernet link. Fig. 2 shows the System-on-Chip (SoC) module with main IP cores and connections next to the companion computer.

A. Hardware modules

We extended and adapted a visual-inertial sensor as described in [20] to support eight cameras. The cameras are built with MT9V034 image sensors with global shutter from Aptina. The SoC system is based on a Xilinx Zync-7020 SoC module including an ARM dual-core Cortex A9 next to a Xilinx Artix FPGA fabric. The FPGA is interfacing an ADIS 16448 IMU, which is hardware time-synchronized with the triggered image acquisition.

The companion computer is based on an AscTec Mastermind hosting an Intel i7-3612QE quad core CPU with 2.1 GHz and 4 GB of memory.

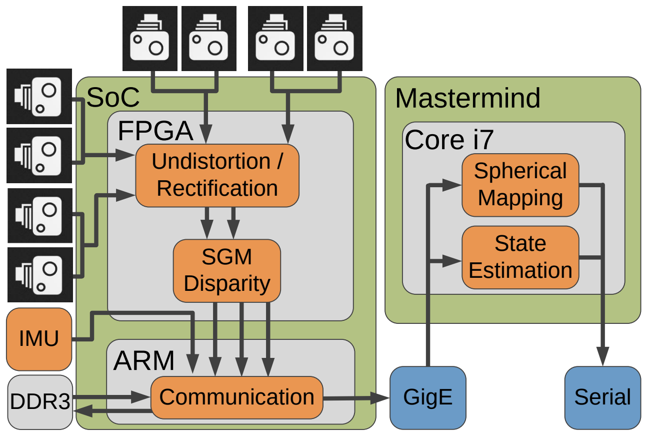

Fig. 2: System overview. Eight image sensors forming four stereo heads are directly connected to the SoC device. Lens distortion correction, rectification and SGM stereo matching is performed within the FPGA fabric. Processed image data and disparity maps are sent to the companion computer using a Gigabit Ethernet link. Spherical mapping is performed on the companion computer and potential obstacles finally sent out using a low latency serial link.

B. FPGA IP Cores

The image sensors are connected to independent deserializer modules first. External triggering maintains synchronization and readout of the two image sensors of a stereo head. The subsequent lens distortion correction and rectification core performs real-time address generation based on provided radial tangential distortion parameters and a rectification homography. Corrected pixel values are sent to the disparity estimation core performing real-time SGM stereo matching as described in [16]. All IP cores are time-multiplexed among the four stereo heads. Calibration parameters are automatically loaded depending on which input stream presents data. The pipeline supports a throughput of up to 80 stereo fps at a resolution of 752x480 pixel each, that can be flexibly distributed among the four stereo heads. Disparity maps are stored next to corrected image data on the shared DDR3 memory of the SoC device. A Linux system running on the ARM cores sends out data over Gigabit Ethernet to the companion computer.

This setup allows us to flexibly select the frequency of each camera pair at run-time, and which image and dense data should be transmitted over the Ethernet link.

C. MAV

An AscTec Firefly platform is used as a carrier for the multi camera system. The Firefly is 605 x 665 x 165 mm in size and 1 kg in weight. The battery allows flights up to 12 minutes and a payload of up to 600 g. The available onboard computer AscTec Mastermind weights 325 g and our camera setup another 235 g, which brings us close to the limits. Fig. 3 shows the Firefly MAV platform with the mounted eight camera system.

D. Calibration

To perform rectification, undistortion of the stereo images and reprojection of the resulting disparity map, the intrinsic

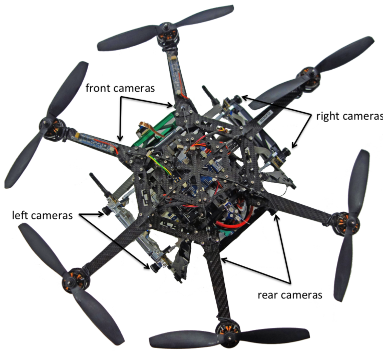

Fig. 3: AscTec Firefly platform with mounted eight camera setup. The four stereo camera heads are targeted towards different directions to cover the surroundings of the MAV.

Device	**Operation**	**Time** (us)	**Cumulative Latency** (us)
CMOS Sensor	Image Capture	3000 us	3000 us
FPGA	Undistort./Rectification	1200 us	4200 us
	SGM Stereo	300 us	4500 us
Core i7 3612	Data transmission	14'000 us	18'500 us
	Speckles Filtering	7'200 us	25'700 us
	Obstacle Mapping	1'200 us	26'900 us
	Map Moving*	29'600 us	26'900 us

TABLE I: Table of timings of individual parts of the algorithm and overall latency. The timings start at the beginning of image exposure and end after determination of close obstacles. The run times on the CPU core are subject to variation, the maximum times are mentioned. *Map Moving is done in a separate thread and does not cause any extra latency.

and extrinsic calibration parameters of the individual stereo camera pairs are required to be known. Additionally, to build a locally consistent map as well as to perform state estimation, the extrinsic parameters of the stereo pairs relative to each other and relative to the IMU are required. All calibration parameters are estimated using the Kalibr framework [21].

V. Results

In this section, we first show a detailed latency analysis of the algorithms implemented in the test system. We then present detection results in a dynamic scene with persons passing by. Finally, we give field trial results of a test flight in a cluttered outdoor forest scene.

A. Latency Analysis

The timing of the individual steps of the algorithm are shown in Table I. The timings start at image exposure and end when the obstacle is being sent to a potential collision avoidance system. A detailed latency analysis of the FPGA IP cores is given in [16]. The transfer of the disparity maps and raw images to the companion computer using an Ethernet link results in the largest portion of the overall latency.

B. Dynamic Scene

We placed the MAV in an outdoor open space while persons walked by. A 3D point cloud of the spherical map with data inserted from all four disparity maps is shown in Fig. 1. The spherical map representation reduces the total amount of data and allows for real-time obstacle detection. Fig. 4 shows the disparity map of the left and the front camera head next to a top down view of the spherical map with labeled obstacles. For this experiment, we recorded dense data in all four directions with 7 Hz each. To avoid collisions in FPGA resources and congestion in the data transmission, we evenly distributed the four dense triggers over one phase. This results in a round robin map update schedule, which minimizes delays caused by processing time needed of the map update step.

C. Field Trial

We verified the system in outdoor flight tests. The MAV successfully detected potential obstacles in a forest environment while updating the spherical map. The sensor rate is set to 21 Hz for the two frontal raw image streams for state estimation and all four disparity map streams to 7 Hz.

In Fig. 5a, the rectified left camera view of the front stereo head is presented next to the corresponding disparity map in Fig. 5b. The 3D spherical map with the MAV coordinate system is presented in Fig. 5g. The environment is color-coded according to the distance of the objects towards the MAV (red: close, blue: distant). While the MAV moves forward, obstacles are getting closer and pass by on the side and enter the visible area of the sideways looking stereo heads. The tree trunk visible in Fig. 5b shows up in Fig. 5h at the side of the MAV and disappears in this viewpoint in Fig. 5i. This paper is accompanied by a video showing the flight tests and real-time spherical map building.

VI. Conclusion

In this work, we presented a low-latency FPGA-based omnidirectional obstacle detection sensor system. By integrating this sensor setup on a MAV platform, we demonstrate successful obstacle detection with very little latency. Our mapping algorithm outperforms existing dense omnidirectional sensing solutions in terms of latency due to the stream-based FPGA data processing pipeline. Low-latency obstacle detection around the MAV allows for usage in collision avoidance in highly cluttered dynamic environments.
Due to the FPGA, a multi camera setup is feasible without heavily upgrading the overall computational power. The perception system including FPGA, image sensors and stereo mounts is 235 g in weight, thus rendering it a highly interesting sensor solution for mobile robotic systems.

References

[1] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous obstacle avoidance and maneuvering on a vision-guided mav using on-board processing," in *Proceedings of ICRA*, 2011.
[2] F. Andert, F. Adolf, L. Goormann, and J. Dittrich, "Mapping and path planning in complex environments: An obstacle avoidance approach for an unmanned helicopter," in *Proceedings of ICRA*, IEEE, 2011.
[3] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *International Symposium on Robotics Research (ISRR)*, 2011.
[4] A. Bachrach, S. Prentice, R. He, P. Henry, A. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Estimation, planning, and mapping for autonomous flight using an rgb-d camera in gps-denied environments," *The International Journal of Robotics Research*, 2012.

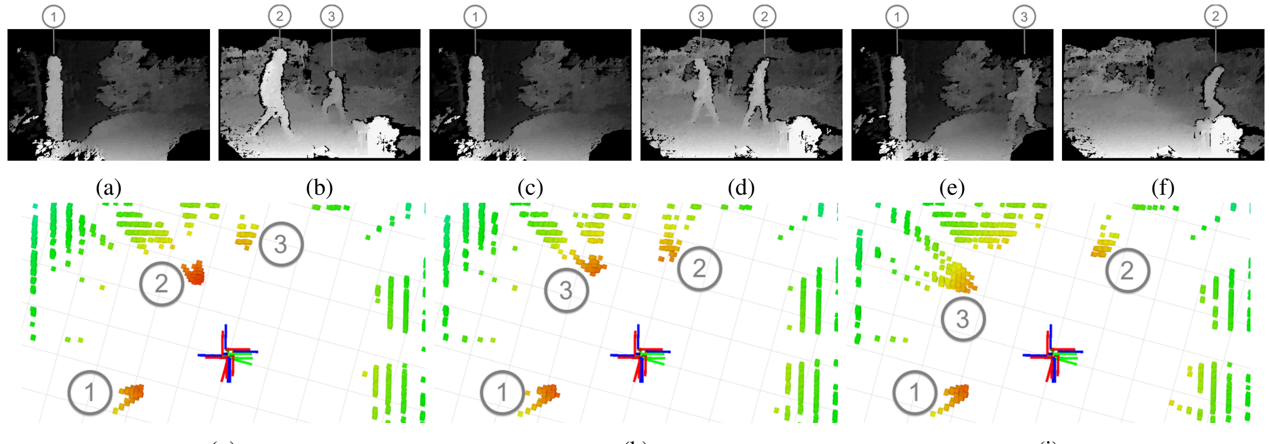

Fig. 4: Disparity maps of left (a, c, e) and front (b, d, f) camera head for different time steps and an horizontal cutout of the according spherical map in top down view with labeled obstacles in (g-i)

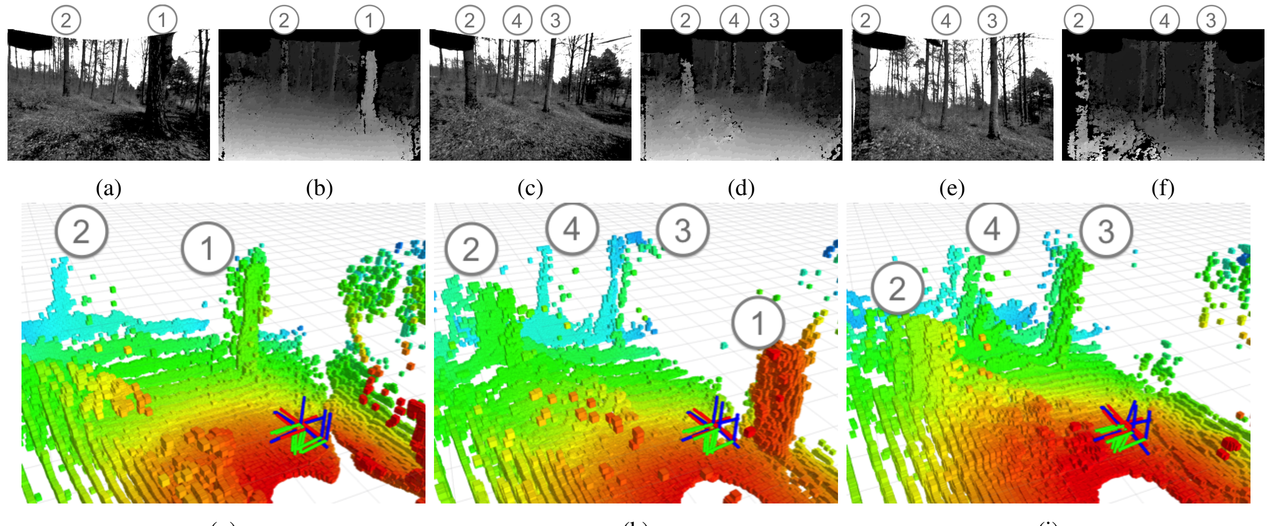

Fig. 5: Left rectified image and disparity map of the front camera head for different time steps in (a-f). The resulting spherical map representation in perspective view with labeled obstacles in (g-i).

[5] H. Oleynikova, D. Honegger, and M. Pollefeys, “Reactive avoidance using embedded stereo vision for MAV flight,” in *Proceedings of ICRA*, 2015.

[6] A. J. Barry and R. Tedrake, “Pushbroom stereo for high-speed navigation in cluttered environments,” in *Proceedings of ICRA*, 2015.

[7] S. Omari, P. Gohl, M. Burri, M. Achtelik, and R. Siegwart, “Visual industrial inspection using aerial robots,” in *Proceedings of CARPI*, 2014.

[8] D. Holz, M. Nieuwenhuisen, D. Droeschel, M. Schreiber, and S. Behnke, “Towards multimodal omnidirectional obstacle detection for autonomous unmanned aerial vehicles,” *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-1/W2, pp. 201–206, 2013.

[9] H. Koyasu, J. Miura, and Y. Shirai, “Real-time omnidirectional stereo for obstacle detection and tracking in dynamic environments,” in *Proceedings of IROS*, 2001.

[10] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “OctoMap: An efficient probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, 2013.

[11] R. Brockers, Y. Kuwata, S. Weiss, and L. Matthies, “Micro air vehicle autonomous obstacle avoidance from stereo-vision,” in *SPIE Defense + Security*, 2014.

[12] C. Pantilie and S. Nedevschi, “Real-time obstacle detection in complex scenarios using dense stereo vision and optical flow,” in *Proceedings of ITSC*, 2010.

[13] S. Omari, M. Bloesch, P. Gohl, and R. Siegwart, “Dense visual-inertial navigation system for mobile robots,” in *Proceedings of ICRA*, 2015.

[14] S. Vanneste, B. Bellekens, and M. Weyn, “3DVFH+: Real-Time Three-Dimensional Obstacle Avoidance Using an Octomap,” *researchgate.net*.

[15] J. Borenstein and Y. Koren, “The Vector Field Histogram-Fast Obstacle A voidance for Mobile Robots,” *IEEE Transactions on Robotics and Automation*, 1991.

[16] D. Honegger, H. Oleynikova, and M. Pollefeys, “Real-time and low latency embedded computer vision hardware based on a combination of fpga and mobile cpu,” in *Proceedings of IROS*, 2014.

[17] D. Brown, “Decentering distortion of lenses,” *Photogrammetric Engineering*, no. 32, pp. 444–462, 1966.

[18] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual-inertial odometry using nonlinear optimization,” *The International Journal of Robotics Research*, 2014.

[19] G. Bradski *Dr. Dobb’s Journal of Software Tools*.

[20] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart, “A synchronized visual-inertial sensor system with fpga pre-processing for accurate real-time slam,” in *Proceedings of ICRA*, 2014.

[21] P. Furgale, J. Rehder, and R. Siegwart, “Unified temporal and spatial calibration for multi-sensor systems,” in *Proceedings of IROS*, 2013.