



Eckehard Steinbach, Matthias Kranz, Werner Maier, Florian Schweiger, Nicolas Alt, Anas Al-Nuaimi, Clemens Schuwerk (Hrsg.)

ADVANCES IN MEDIA TECHNOLOGY

VISUELLE UMGBUNGSMODELLIERUNG FÜR ROBOTIK- UND
SPIELEANWENDUNGEN

21. Juni 2011

ISSN 2191-8015

Technische Universität München
Lehrstuhl für Medientechnik

VISUELLE UMGEBUNGSMODELLIERUNG FÜR ROBOTIK- UND SPIELEANWENDUNGEN

Bei modernen Robotern, wie z.B. dem PR-2 von der Firma Willow Garage, spielt das Erstellen von Umgebungsmodellen eine wichtige Rolle, um Objekte, die für ihre Aufgaben relevant sind, zuverlässig erkennen und manipulieren zu können. Unter Verwendung von mehreren Kameras ist der Roboter in der Lage, Informationen über die Geometrie seiner Umgebung zu erhalten, welche zur Navigation verwendet werden können. Ein wichtiger Aspekt ist in diesem Zusammenhang auch die Selbstlokalisierung des Roboters in seiner Umgebung, die oft zusammen mit der Erstellung der Umgebungskarte durchgeführt wird (Visual SLAM).

Auch bei modernen Spielekonsolen, wie z.B. der XBox von der Firma Microsoft, ist die Modellierung der Umgebung von großer Bedeutung. Die dabei verwendete Kinect-Kamera bietet die Möglichkeit, sowohl Bild- als auch Tiefeninformation in Echtzeit aufzunehmen, wodurch die Spieler als Teil der virtuellen Welt des Spiels in Szene gesetzt werden können. Die Spieler interagieren mit Gesten und Körperbewegungen mit der virtuellen Welt, wodurch sie den Eindruck erhalten, vollständig in die virtuelle Welt einzutauchen.

In diesem Sammelband werden Algorithmen und Verfahren beschrieben, die mit der Umgebungsmodellierung aus Kameradaten im Hinblick auf Robotik- und Spieleanwendungen in Zusammenhang stehen. Die Ausarbeitungen wurden von den Teilnehmern des Hauptseminars Medientechnik im Sommersemester 2011 verfasst.

Das Seminar wurde vom Lehrstuhl für Medientechnik an der Technischen Universität München durchgeführt.

Dieser Sammelband enthält die folgenden Themen:

1. The Kinect Sensor Platform
2. Visual Geometric Environment Modeling for Augmented Reality and Robotics
3. Visuelle Navigation basierend auf einer topologischen Bildersammlung als Umgebungsmodell
4. Automatische Erstellung von 3D-Objektmodellen mit Hilfe von Tiefeninformationen aus Kamerabildern
5. Semantisch bedeutungsvolle Modelle für Visual SLAM

Die Webseite mit der elektronischen Version dieses Sammelbandes und zusätzlichen Informationen findet sich unter http://www.lmt.ei.tum.de/courses/hsmt/hsmt.2011_SS.html

Wir danken allen Studenten für ihre Mühe und ihre Teilnahme am Seminar!

München, Juni 2011

Prof. Dr.-Ing. Eckehard Steinbach
Prof. Dr. Matthias Kranz
Werner Maier
Florian Schweiger
Nicolas Alt
Anas Al-Nuaimi
Clemens Schuwerk
(Hrsg.)

INHALTSVERZEICHNIS

The Kinect Sensor Platform (Thomas Kühn)	1
Visual Geometric Environment Modeling for Augmented Reality and Robotics (Gabriel Schönung)	5
Visuelle Navigation basierend auf einer topologischen Bildersammlung als Umgebungsmodell (Wolfgang Bremer)	9
Automatische Erstellung von 3D-Objektmodellen mit Hilfe von Tiefeninformationen aus Kamerabildern (Alma Pröbstl)	13
Semantisch bedeutungsvolle Modelle für Visual SLAM (Alexander Schreiber)	17

The Kinect Sensor Platform

Thomas Kühn
 Technische Universität München
 thomas.kuehn@mytum.de

Abstract

Within this work the basic concepts of the Kinect Sensor platform are presented and analyzed. The pinhole camera model and sensor calibration are introduced to achieve a basic understanding of computer vision. Then structured light and the specific Kinect pattern are examined. The key feature depth estimation is analyzed and sensor fusion between the IR and RGB camera is covered.

1. Introduction

There are different kinds of approaches to control a machine like a PC or a gaming device, classic ones being keyboard, mouse, or controller. Microsoft's Kinect forms a whole new category and was originally developed to control games on the Xbox 360 without having the need of holding a device like a controller. High accuracy at low cost is the most important incentive, resulting in high popularity and adoption even to robotics. "Hackers" wrote an open source driver called libfreenect to connect and use the Kinect with a computer instead of its designated use with the Xbox. In addition PrimeSense, who has developed the reference design, began to supply the OpenNI driver and the NITE Middleware software for scene perception. Due to the success of the Kinect, Microsoft recently released an SDK (beta) to officially access Kinect's capabilities on a PC [5].

In chapter 2, we will discuss the design of the Kinect, cover camera models, and calibration. Additionally, the concept of structured light and specific pattern are discussed. The analysis is completed by looking into depth calculation and sensor fusion. Finally in chapter 3 conclusions are drawn.

2. Kinect

2.1 Design of the Kinect

The Kinect consists of three optical components: a laser-based infrared (IR) projector, an IR camera and a color camera. A specific characteristic of the Kinect is the combined

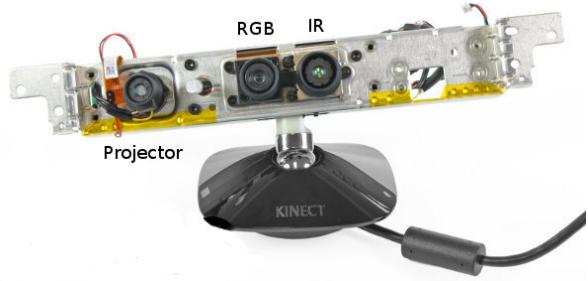


Fig. 1: Inside the Kinect [4]

projector and camera, instead of the usual two cameras for stereo vision. The key advantage of this solution is reduced computation.

For the audio part there is a multiarray microphone, which consists of four sensors and is able to separate sound from different directions. Within this work, the focus is on the visual part and sound is not considered further.

In Figure 1 it can be seen that the IR projector is on the very left and the RGB and IR cameras are very close to each other in the middle.

2.2 Sensor Calibration

Both cameras and the IR projector can be approximated with the classical pinhole model shown in Figure 2. A real world point $\mathbf{X} = (X, Y, Z)^T$ is mapped to $\mathbf{x} = (fX/Z, fY/Z, f)^T$ into the image plane, at the intersection of a line from the camera centre \mathbf{C} to \mathbf{X} . Ignoring the final coordinate, a mapping from Euclidean \mathbb{R}^3 space to \mathbb{R}^2 is achieved with $(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T$.

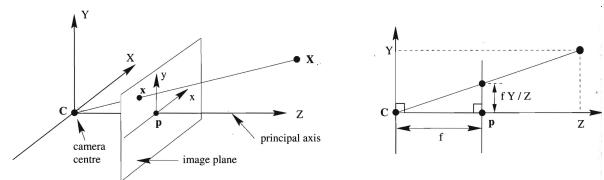


Fig. 2: Pinhole Camera Model [2]

From now on homogenous coordinates will be used such that a 3D point in space is described by $\mathbf{X} = (X, Y, Z, 1)^T$. The equivalent image point is represented by $\mathbf{x} = (x, y, 1)^T$.

According to [2] and [7], the relationship between a 3D point in space and its projection is given then by

$$\mathbf{x} = \mathbf{K} [\mathbf{R} \ t] \mathbf{X} \quad \text{with} \quad \mathbf{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}. \quad (1)$$

\mathbf{K} is called the camera intrinsic matrix containing the principal point coordinates $\mathbf{p} = (x_0, y_0)$, the scale factors in terms of pixel dimensions α_x and α_y in x and y direction, and the skew s of the two image axes. \mathbf{p} is the origin of the image coordinate system and is located on a line through \mathbf{C} perpendicular to the image plane, see Figure 2. $[\mathbf{R} \ t]$ are called extrinsic parameters, which relate real world observations to the camera coordinate frame by the use of a translation and rotation. In Euclidean coordinates a rotation and a translation would imply a multiplication and an addition. Using homogenous coordinates these two operations result in a simple linear operation.

In [7] the model plane is assumed on $Z = 0$ of the world coordinate system. With \mathbf{R} expressed by its columns \mathbf{r}_i , (1) leads to

$$\mathbf{x} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{r}_3 \ t] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ t] \underbrace{\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}}_{\tilde{\mathbf{X}}}. \quad (2)$$

The new vector $\tilde{\mathbf{X}} = (X, Y, 1)^T$ is introduced, since Z is always zero. This reduces complexity in the equations without loss of generality.

The real world point $\tilde{\mathbf{X}}$ can be mapped to its image point by a homography \mathbf{H} with

$$\mathbf{x} = \mathbf{H} \tilde{\mathbf{X}} \quad \text{with} \quad \mathbf{H} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ t]. \quad (2)$$

Furthermore the homography can be denoted $\mathbf{H} = [\mathbf{h}_1 \ \mathbf{h}_2 \ \mathbf{h}_3]$ and from (2)

$$[\mathbf{h}_1 \ \mathbf{h}_2 \ \mathbf{h}_3] = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ t] \quad (3)$$

can be derived. Since \mathbf{r}_1 and \mathbf{r}_2 are the columns of a rotation matrix, they have to be orthonormal, which means $\mathbf{r}_1^T \cdot \mathbf{r}_2 = 0$ and $\|\mathbf{r}_1\|^2 = \|\mathbf{r}_2\|^2$. Therefore (3) can be rewritten as

$$\mathbf{r}_1^T \cdot \mathbf{r}_2 = \mathbf{h}_1^T \mathbf{K}^{-T} \mathbf{K}^{-1} \mathbf{h}_2 = 0 \quad (4a)$$

$$\underbrace{\mathbf{h}_1^T \mathbf{K}^{-T} \mathbf{K}^{-1} \mathbf{h}_1}_{\mathbf{r}_1^T \mathbf{r}_1 = \|\mathbf{r}_1\|^2} = \underbrace{\mathbf{h}_2^T \mathbf{K}^{-T} \mathbf{K}^{-1} \mathbf{h}_2}_{\|\mathbf{r}_2\|^2}. \quad (4b)$$

X	Y	Z
-0.0254	-0.00013	-0.00218

Tab. 1: Typical Translation Values in m [4]

Both equations are the basic constraints on the intrinsic parameters. The determination of the homography \mathbf{H} requires at least four points, because \mathbf{H} has eight degrees of freedom. Therefore a chessboard with more than four points is used to provide more robustness. To determine \mathbf{K} at least three different images with different angles and positions are needed, since \mathbf{K} has five degrees of freedom and every equation in (4) represents one constraint. If pixels of the camera are square, α_x and α_y could be replaced by the focal length f and the minimum image number would be reduced to two, see [7]. However, normally five or more pictures are taken to increase robustness.

Finally, \mathbf{K} can be obtained from $\mathbf{K}^{-T} \mathbf{K}^{-1}$ by applying Cholesky factorization, which is described in detail in [2]. Subsequently, the extrinsic parameters $[\mathbf{R} \ t]$ can be obtained from (2) using the relation

$$[\mathbf{r}_1 \ \mathbf{r}_2 \ t] = \mathbf{K}^{-1} \mathbf{H}.$$

The absolute value of the third rotational vector \mathbf{r}_3 can be calculated by the cross product $\mathbf{r}_1 \times \mathbf{r}_2$. The sign is determined by comparison if the image would be in front or behind the camera.

Applying this method, first the IR and the VGA camera are calibrated separately, and from there the transformation between the two cameras can be found. For the IR projector and the IR camera the approach is based on the same principles described above, with the difference that the projected pattern replaces the checkerboard.

The baseline between the RGB and IR cameras is about 2.5 cm and can be seen in Figure 1. Typical translation values from calibration are shown in Table 1 and the baseline (X) is in good accordance to the measurement. Besides, the rotation component was also very small with offsets about 0.5° [4]. This translates to an offset of 1 cm at a distance of 1.5 m.

Reprojection errors mainly due to lens-distortion and de-centering can be further reduced in an iterative optimization over all involved parameters. Typical reprojection errors for several Kinect devices are in the order of 0.1 to 0.2 pixels and are shown in Table 2. This concludes that neither the IR or the RGB camera were calibrated for distortion correction, but the assembly is of high quality. Microsoft supplies a calibration card containing certain shapes and colors to perform fine calibration of an Xbox 360 at home [3]. This card has a rectangle on it, which provides four easily detectable corner points, in accordance

	original	calibrated
IR	0.34	0.17
RGB	0.53	0.16

Tab. 2: Typical Projection Errors in pixel [4]

with the above mentioned constraints on the homography \mathbf{H} .

2.3 Structured Light and Pattern

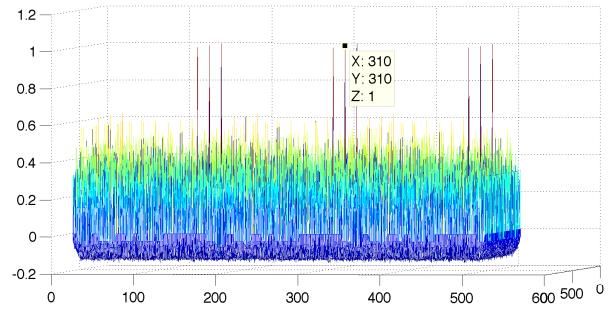
In [1] three kinds of invisible structured light are described: Filtered Structured Light, Imperceptible Structured Light and InfraRed Structured Light (IRSL). Advantages of using structured light are the absence of sending an invasive signal, while having a precise and reliable 3D reconstruction. For the Kinect IRSL is used because of the highest resolution and accuracy of all three techniques.

The IR projector sends out a speckle pattern, which is used for depth estimation, see 2.4, and is constant over time. The PrimeSense's pattern is investigated for replications in [6]. The pattern repeats itself after 211 horizontal spots and 165 vertical spots and in every of these blocks there is a bright center point. The total pattern is composed of a 3x3 repetition of the before mentioned spots, which results in 633x495 spots. This repetition has the disadvantage that disparity has to be limited to one third of pixels in each direction. In practice this is irrelevant, because disparities are limited to 64 neighboring pixels, see 2.4.

A curiosity about the pattern is the 180° invariance, which has probably no algorithmical advantage. The author supposes this might have some practical advantages, so the projector can be mounted upside down, which increases the security against mistakes during production.

To confirm the arrangement of the speckle pattern, Matlab computations were made to find correspondences. Different pictures of the pattern, taken with consumer cameras, and a window size of 9x9 were used to find correlations in the picture. The correlation template was selected at 310x310 and can be detected at this position. Furthermore, due to the replication of the pattern another 8 peaks can be found, which is shown in Figure 3.

These expected 9 peaks could not be found in all analyzed pictures. This is probably due to distortion from the camera optics, from the configuration how the picture was taken, and from pin-cushion distortion of the pattern.

**Fig. 3: Correspondences Calculation with Matlab Using a 9x9 Correlation Template**

2.4 Depth Estimation

Normally a stereo camera system is used, which is rectified, this means to have parallel images, perpendicular to the same baseline. The disparity is typically calculated as the x-axis difference on the two cameras. The Kinect consists of a camera for 3D vision and a projector with a specific pattern, which is memorized. The local pattern at each pixel can be compared to the memorized one, e.g. using cross-correlation at this pixel and supposedly 64 neighboring pixels [4]. Considering the best match there is a certain disparity d , from which the pixel depth can be calculated. To achieve a rectified system, (1) can be first rewritten as

$$\mathbf{x} = \mathbf{KR} [1 \mid -\mathbf{C}] \mathbf{X}.$$

Here, \mathbf{C} is the position of the projection center expressed in the world coordinate frame. \mathbf{K}' and \mathbf{R}' describe the internal and extrinsic parameters of the rectified camera and projector and the rectification is achieved by

$$\mathbf{K}'\mathbf{R}'\mathbf{R}^T\mathbf{K}^{-1} \cdot \mathbf{x} = \mathbf{K}'\mathbf{R}' [1 \mid -\mathbf{C}] \mathbf{X}.$$

Then the depth z can be calculated as in [4] using the relation

$$z = b \cdot f/d,$$

with the horizontal baseline b (in meters), the (common) focal length f of the cameras (in pixels), and the disparity d (in pixels). Details on the implementation can be found in [4] and are summarized in the remaining part.

The larger the values for the disparity are, the closer are the objects. If the disparity is 0, the rays of the two cameras would be parallel and the object at infinite depth. This is true for a normalized system, but the Kinect does use their own one, so a disparity of 0 does not imply infinite distance. This is due to a shift of the coordinate center of the IR camera, so the raw disparities can be better represented



Fig. 4: Self Taken Depth Image of The Kinect

as integers. The relationship to a normalized disparity can be shown as

$$d = \frac{1}{8} \cdot (d_{\text{off}} - k_d),$$

with the normalized disparity d , the Kinect disparity k_d , the offset d_{off} for a certain device, and $1/8$ is needed due to the Kinect disparity k_d is in $1/8$ pixel units. Typical values are $d_{\text{off}} = 1090$ and $b = 7.5$ cm.

An example of a hand tracking algorithm, using a depth image, can be seen in Figure 4. The tracked hand is marked by a white point and its trajectory. The depth image, which can be returned with the non-interpreted IR image, is calculated on a chip within the Kinect. By inspecting the two images an offset occurs, which is probably the implication of the correlation window size. There is a small 8 pixel black band of the disparity image.

A correlation window defines an area where pixels are compared. Due to the choice of the reference point, some pixels can not be calculated. Assume a 9x9 pixel window with its reference point in the middle, horizontally 4 pixels on the left and 4 pixels on the right of the frame could not be calculated. The same would be true vertically and in total 8 pixels are not able to be determined in each direction.

The IR imager is an Aptina MT9M001 with a resolution of 1280x1024. If a 2x2 binning is used the resulting resolution would be 640x512. The Kinect returns a raw disparity image with VGA resolution (640x480). The mentioned 8 pixel black band on the right make sense if the Kinect has a 9x9 (or 9x7) pixel correlation window and the reference point would be on the left. Vertically the correlation window has no effect since the resolution with 512 pixels is higher than the used 480 pixels.

2.5 Sensor Fusion

The Kinect enables to fuse a colored RGB image with a depth image. With calibrated cameras, as discussed in 2.2, the pixels from the range image, containing depth informa-

tion, are mapped to the color pixels of the RGB frame. The result are pixels, which contain color data and 3D coordinates, and composed form a single image.

3. Conclusions

The Kinect provides an effective approach for 3D vision offered at a very competitive price. Therefore, it became not only very popular in the field of gaming but even in robotics the Kinect is a more and more often used device. This success also benefited from free drivers and software for scene perception.

The strongest competitor is Sony's Move which follows another approach: The player holds a controller with a colored bulb in his hand that is tracked by only one camera. On the one hand the controller enables force feedback, but on the other hand the system is not designed to track a person and create a skeleton model like the Kinect. The Move allows the increase of the frame rate from 60 fps to 120 fps while cutting the resolution by 50 % to 320x240 pixels. The effect is a more immediate response and reduced motion blurring. In contrast to the Kinect, the Move is not as interesting for the field of science, especially in robotics.

For the future, systems with higher camera resolutions and faster image processing could supply the demand for an even higher precision and a more realistic experience.

References

- [1] D. Fofi, T. Sliwa, and Y. Voisin. A comparative survey on invisible structured light. *SPIE Electronic Imaging-Machine Vision Applications in Industrial Inspection XII*, pages 90–97, 2004.
- [2] R. Hartley and A. Zisserman. *Multiple view geometry*, volume 3. Cambridge university press, 2000.
- [3] B. Klug. Microsoft kinect: The anandtech review, 2010. <http://www.anandtech.com/show/4057/microsoft-kinect-the-anandtech-review/> (accessed 10.06.2011).
- [4] K. Konolige and P. Mihelich. Technical description of kinect calibration, 2010. http://www.ros.org/wiki/kinect_calibration/technical (accessed 14.06.2011).
- [5] Microsoft. Kinect for windows sdk, 2011. <http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/> (accessed 18.06.2011).
- [6] A. Reichinger. Kinect pattern uncovered, 2011. <http://azttm.wordpress.com/2011/04/03/kinect-pattern-uncovered/> (accessed 09.06.2011).
- [7] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

Visual geometric environment modeling for augmented reality and robotics

Gabriel Schönung
 Technical University of Munich
 Institute for Media Technology
 Arcisstr. 21, 80333 München, Germany
 gschoenung@mytum.de

Abstract

This paper presents the main steps to create a dense environment model out of a sequence of images. It will concentrate on a single camera based approach. First the camera motion will be estimated in order to create a spare representation of the model points in space. Then implicit surface reconstruction by fitting a function to the 3D point model is applied. Finally the local models are refined through dense correspondence measurements and integrated into one global surface model. This method applies notably to augmented reality and robotics where detailed environment models are required.

1. Introduction

Visual geometric environment models are dense 3D representations of a scene, for example a workspace. The applications of such 3D models are various, ranging from augmented reality games (figure 4) to industrial systems and robotics. Detailed representation are needed in case of interaction with the environment and where a realistic physics engine has to be integrated. A robot requires an exact 3D model of an object to grasp it. Collision avoidance is also an important topic in many applications.

The Microsoft Kinect allows to build 3D representations due to an infrared depth sensor. Here, however, we want to focus on creating a 3D model by using just a single camera. We divided our paper in the main steps that lead from a sequence of taken images from a scene to a global 3D representation (figure 2): In order to create a first model, we have to track characteristic points, so called features, retrieve the camera motion and create a 3D point map out of them. Then we can create a dense representation out of our spare model and fuse the local models of the different parts of the scene into a global one.

The whole method is mainly based on three publications: The structure of the paper is inspired by [8]. In [7], the



Figure 1. Dense reconstruction from a sequence of images. A spare representation of the map is created, local dense models are built and fused into a global surface model. (Image from [10])

camera tracking part of section 2.2 is separated from a mapping part and processed in parallel threads on a multi-core processor. The basic idea of the dense reconstruction (section 2.3) is taken from [10].

2. Method

2.1 Feature detection and correspondence

In order to track a set of features throughout a sequence of images, the first step consists of selecting them. There are different kinds of image features like edges, corners (interest points) or blobs (regions of interests). For this application the most commonly used features are corners, which can be defined as an intersection of two edges. An interest point has a well-defined position in the image and should be detected robustly. To detect corners in an image, the simplest way would be to use correlation. But this is computationally expensive. One of the earliest corner detection algorithms computes the sum of squared differences (SSD) between a patch around a candidate corner and patches shifted nearby in the four main directions. A locally maximal number indicates an interest point. The frequently used Harris corner detection algorithm [3] improves this approach by computing a taylor approximation to the sec-

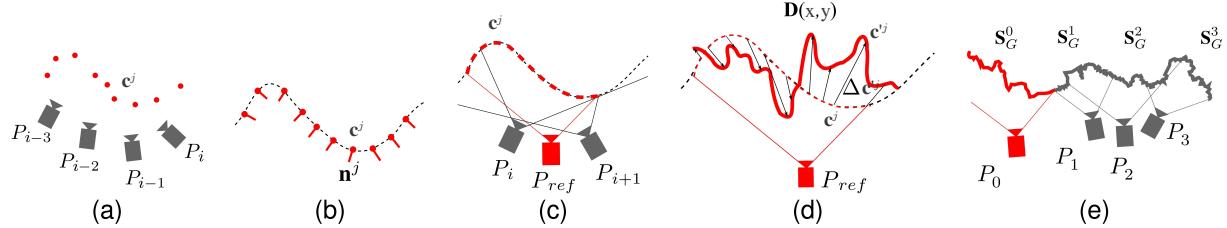


Figure 2. Dense reconstruction pipeline in an overview. (a) The camera motion is estimated and a spare point map is created. (b) A base model is fitted to the points. (c) Parts of the scene are visible in several camera frames. (d) The base model is refined through an iterative algorithm. (e) The local dense representations are integrated into one global model. (Image from [10])

ond derivative of the SSD with respect to the shift. There are different variations of corner detectors which are invariant to translations, rotations and uniform rescalings in the spatial domain, notably the Harris-Affine detector [9]. Once the candidate interest points are selected, the plan is to match them across the sequence of images. The simplest way would be to use SSD again between two images in a small window around a feature point [8].

2.2 Projective reconstruction

As a next step we want to retrieve the camera motion and the 3D point map of the scene. We will introduce the camera motion task with a 2D-3D pose estimation. Therefore we suppose for now that an initial 3D point map of the scene has already been acquired. If feature points of the map match with the 2D image, we can apply the POSIT method which finds the rotation matrix and the translation vector of the 3D points [1]. It consists of an iterative algorithm that approximates the perspective projection in every step with a scaled orthographic projection of the feature points.

Usually there is no initial map. Therefore we estimate the essential matrix through image correspondences and find then 3D points with triangulation.

2.2.1 Essential matrix

For the triangulation of the base map, we have to estimate the essential matrix, which relates corresponding points in a stereo camera model. In this case we are taking two consecutive images as the two views. The essential matrix satisfies the epipolar equation:

$$\mathbf{x}_2^T \mathbf{F} \mathbf{x}_1 = 0$$

where \mathbf{F} is the fundamental matrix, $\mathbf{x}_1 = (x_1 \ y_1 \ 1)^T$ are the homogeneous image coordinates of a corresponding point in the first image and $\mathbf{x}_2 = (x_2 \ y_2 \ 1)^T$ in the second image.

If we define $\mathbf{f} = (f_{11} \ f_{12} \ f_{13} \ f_{21} \ f_{22} \ f_{23} \ f_{31} \ f_{32} \ f_{33})^T$ as a vector of the matrix elements of \mathbf{F} and we have n correspondences we get the system of equations $\mathbf{A} * \mathbf{f} = \mathbf{0}$ with:

$$\mathbf{A} = \begin{pmatrix} x_2^1 x_1^1 & x_2^1 y_1^1 & x_2^1 & y_2^1 x_1^1 & y_2^1 y_1^1 & y_2^1 & x_1^1 & y_1^1 & 1 \\ \vdots & \vdots \\ x_2^n x_1^n & x_2^n y_1^n & x_2^n & y_2^n x_1^n & y_2^n y_1^n & y_2^n & x_1^n & y_1^n & 1 \end{pmatrix}$$

Because coordinates of corresponding points satisfy the epipolar equation from above, the columns of \mathbf{A} are linearly dependent. At best the matrix has maximal rank 8. The matrix \mathbf{F} determined out of \mathbf{f} could be non-singular because of measurement errors. A well-known approach to this problem is the eight-point algorithm [4].

2.2.2 Triangulation

After having found the essential matrix, we can compute the camera matrices of the two images as described in [6, result 9.14]. Given the camera matrices \mathbf{P}_1 and \mathbf{P}_2 and the coordinates of \mathbf{x}_1 and \mathbf{x}_2 of a corresponding point in both images, the two rays and the intersecting point in space \mathbf{c} may easily be computed (figure 3).

In presence of noise, the two rays will not generally meet. Several methods to this problem are discussed in [5].

2.2.3 Adding new keyframes

Initially the map is created of only two keyframes. When the camera moves, new keyframes will be added to the system. A computational method to estimate a projective construction from a number of frames is bundle adjustment: Given a set of image coordinates \mathbf{x}_i^j (of the j -th point in the i -th keyframe), we want to find the set of corresponding camera matrices \mathbf{P}_i and the 3D points \mathbf{c}^j such that:

$$\mathbf{P}_i \mathbf{c}^j = \mathbf{x}_i^j$$

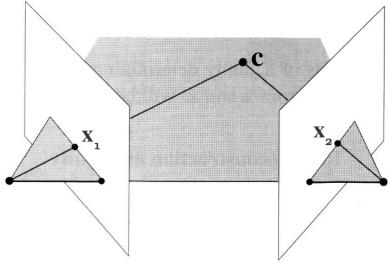


Figure 3. Triangulation. The two rays lie in a plane and intersect in a point c in space. (Image from [8]).

As the image could be noisy, this equation might not be satisfied. With Levenberg-Marquardt bundle adjustment the map can iteratively be adjusted by minimizing the re-projection error ([7, section 6.3]).

2.3 Dense reconstruction

In the following we will pick up the steps proposed by Newcombe *et al.* [10]. First, an approximate base surface is computed out of the spare representation from the previous sections to get a dense but approximated model of the scene. In a next step, this base model is refined and finally the different dense reconstructions are integrated into one global surface model.

2.3.1 Base surface reconstruction

To get a first approximate dense representation, we could for example fit Bézier surfaces into our spare point model [2]. A Bézier surface of order one can be defined as a parametric surface where the coordinates s and t of a point \mathbf{Q} are given by :

$$\mathbf{Q}(s, t) = \sum_{k=0}^3 \sum_{l=0}^3 B_k(s) B_l(t) \mathbf{P}_{k,l}$$

where $B_m^n(u)$ is a Bernstein polynomial:

$$B_m^n(u) = \binom{n}{m} u^m (1-u)^{n-m}$$

and $\mathbf{P}_{k,l}$ is the set of our model points.

This method is computationally expensive. A more advanced and faster technique would be a hierarchical approach to 3D scattered data interpolation with compactly supported basis functions [11].

2.3.2 Refining the base model

After having created a base surface, the depth map needs to be refined through an iterative algorithm. Every visible surface point $\mathbf{c}^j = [c_x^j \ c_y^j \ c_z^j]^T$ has an image projection $\mathbf{x}_i^j = (x_i^j \ y_i^j)^T = \mathbf{P}_i(\mathbf{x}^j)$ in every camera frame i . A small displacement of a surface point in space $\Delta\mathbf{c}^j = \mathbf{c}'^j - \mathbf{c}^j$ results in a displacement of the corresponding projections $\Delta\mathbf{x}_i^j = \mathbf{x}'_i^j - \mathbf{x}_i^j$. This can be approximated by the so called scene flow:

$$\Delta\mathbf{x}_i^j = \mathbf{J}_i^{c^j} \Delta\mathbf{c}^j$$

where $\mathbf{J}_i^{c^j}$ is the Jacobian of the projection function evaluated at point \mathbf{c}^j . As the base mesh provides a prediction of the position (and normal) of the visible surface elements, this information can be used in order to compute a more accurate estimation of \mathbf{c}^j . Assuming that the displacement $\Delta\mathbf{x}_i^j$ can be measured, the Jacobian can be computed and the over-constrained linear system for $\Delta\mathbf{c}^j$ can be solved. In a reference frame \mathbf{x}_{ref}^j , a ray for every pixel is intersected with the base model to determine a position \mathbf{c}^j in space (figure 5). Then this position is projected into every other frame in order to obtain a predicted image correspondence \mathbf{x}_i^j . The scene flow vector $\Delta\mathbf{c}_i^j$ points from \mathbf{c}^j to \mathbf{x}_{ref}^j and can be described as:

$$\Delta\mathbf{c}^j = \lambda^j \mathbf{r}^j$$

where \mathbf{r}^j is the unit vector in direction of the ray:

$$\mathbf{r}^j = \frac{\mathbf{R}_{cw}^{ref} [\mathbf{x}_{ref}^j | 1]}{\|\mathbf{R}_{cw}^{ref} [\mathbf{x}_{ref}^j | 1]\|}$$

and \mathbf{R}_{cw}^{ref} is the rotation matrix from the camera into the world coordinate system.

That brings us to a overdetermined equation system:

$$\Delta\mathbf{x}^j = \begin{bmatrix} x_1^j \\ y_1^j \\ \vdots \\ x_n^j \\ y_n^j \end{bmatrix} = \mathbf{J}^{c^j} \mathbf{r}^j \lambda^j$$

where $\Delta\mathbf{x}_i^j = \mathbf{x}'_i^j - \mathbf{x}_i^j$ is the displacement for each frame $i = 1 \dots n$. This can be solved through least squares $\min_{\lambda^j} \|\mathbf{J}^{c^j} \mathbf{r}^j \lambda^j - \Delta\mathbf{x}^j\|$ to get λ^j .

As a result a dense update from our base surface is obtained:

$$\mathbf{c}'^j = \mathbf{c}^j + \mathbf{r}^j \lambda^j$$

2.3.3 Local model integration

As a last step, the local dense models of our scene have to be integrated into one global representation.

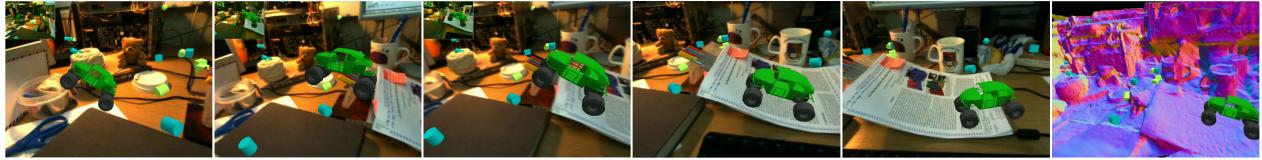


Figure 4. Use of dense reconstruction for augmented reality game. Due to the detailed dense environment model, a realistic physics engine can be implemented. The car is performing a jump. On the right is a rendered 3D model of the scene. (Image from [10])

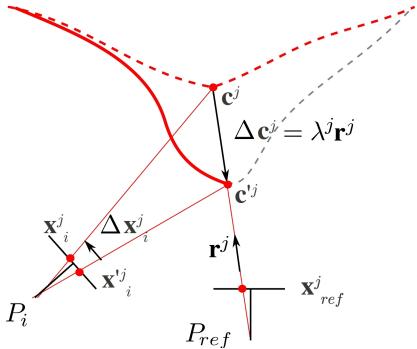


Figure 5. The geometry of refining the base model. The refining occurs on the viewing ray between the 3D point and a reference frame. (Image from [10])

A simple method would be to remove vertices in the new model that are below a bound ϵ_{dist} . In [10], they also remove less visible vertices with a big error in the new mesh. However, there are more developed methods to solve this problem. Zach [12] for example is applying a volumetric range image integration in his GPU-accelerated implementation to return very accurate models.

3. Results

All results are taken from Newcombe *et al.* [10] and are obtained with a hand-held camera with 640 x 480 resolution. The computation was performed on a quad-core PC. Besides the images in this paper (figure 1 and 4), there are more results on Newcombe's website (<http://www.doc.ic.ac.uk/~rnewcomb/CVPR2010/>) including a video.

4. Conclusions

This paper presents a step-by-step approach to create dense and accurate environment models out of a sequence

of images taken by a single camera. The shown components are state-of-the-art methods in computer vision and can run on real-time systems. Therefore they apply especially to mobile augmented reality applications and mobile robot platforms where interaction and manipulation of the environment is frequent.

References

- [1] D. F. DeMenthon and L. S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, pages 123–141, 1995.
- [2] G. Farin. *Curves and Surfaces for CAGD*, chapter The Bernstein Form of a Bezier Curve, pages 57–81. Academic Press, fifth edition, 2002.
- [3] C. Harris and M. Stephens. A combined corner and edge detector. *Alvey Vision Conference*, pages 147–151, 1988.
- [4] R. I. Hartley. In defense of the eight-point algorithm. *IEEE Transaction on Pattern Recognition and Machine Intelligence*, 1997.
- [5] R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.
- [6] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.
- [7] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. *6th IEEE and ACM International Symposium on In Mixed and Augmented Reality*, 2007.
- [8] Y. Ma, S. Soatto, J. Košecká, and S. S. Sastry. *An Invitation to 3-D Vision*, chapter Step-by-Step Building of a 3-D Model from Images, pages 375–411. Springer, 2004.
- [9] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.
- [10] R. A. Newcombe and A. J. Davison. Live dense reconstruction with a single moving camera. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010)*, pages 1498–1505, 2010.
- [11] Y. Ohtake, A. Belyaev, and H.-P. Seidel. A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. *Proceedings of Shape Modeling International*, 2003.
- [12] C. Zach. Fast and high quality fusion of depth maps. *Proceedings of the International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008.

Visuelle Navigation basierend auf einer topologischen Bildersammlung als Umgebungsmodell

Wolfgang Bremer, Anas Al-Nuaimi

Technische Universität München

Lehrstuhl für Medientechnik

Arcisstr. 21 80290 München

[wolfgang.bremer, anas.alnuaimi]@tum.de

Zusammenfassung

Das Ziel ist es, eine Methode zu finden, mit der sich ein Roboter nur aufgrund von Bilddaten in einer unbekannten Umgebung bewegen kann. Hierfür wird zuerst der Scalable Vocabulary Tree Algorithmus [6] vorgestellt, der effizient und echtzeitfähig in einer großen Bilddatenbank suchen kann. Darauf aufbauend soll dann erklärt werden, wie die Navigation allein anhand von Bildern [2] möglich ist.

1. Einleitung

Seit Jahren gibt es verschiedene Ansätze um mit Robotern durch unbekannte Umgebungen zu navigieren. Hierfür werden in der Regel verschiedene on- und offboard Sensoren zur Positions- und Lageerkennung eingesetzt. Bekannte Vertreter sind *Light detection and ranging* (LIDAR), Sonar und das *Global Positioning System* (GPS). Mit der wachsenden Computing Power der Roboter werden auch vermehrt bildgebende Verfahren zur Orientierung verwendet. Diese Methode hat verschiedene Vorteile, da mithilfe nur eines relativ billigen Sensors eine Karte erstellt und darin navigiert werden kann. Das größte Problem bei diesem Ansatz bestand bisher darin, in einer großen Menge an Bilddaten die gewünschten ähnlichen oder gleichen Informationen in kürzester Zeit zu finden.

Der hier vorgestellte *Visual Simultaneous Localization and Mapping* (VSLAM) Algorithmus beruht auf einer Methode [2], die 2007 zum ersten Mal vorgestellt wurde. Das darin vorgestellte Verfahren unterscheidet sich von den bisherigen durch die Anwendung des *Scalable Vocabulary Tree* (SVT) [6] als Hilfsmittel zur Suche in großen Bildersammlungen. Dies beschleunigt das *content based image retrieval* (CBIR) in einem großen Datenbestand erheblich. Die folgenden Abschnitte gehen der Reihe nach auf die wichtigsten Bestandteile der Verfahren ein. Zunächst wird

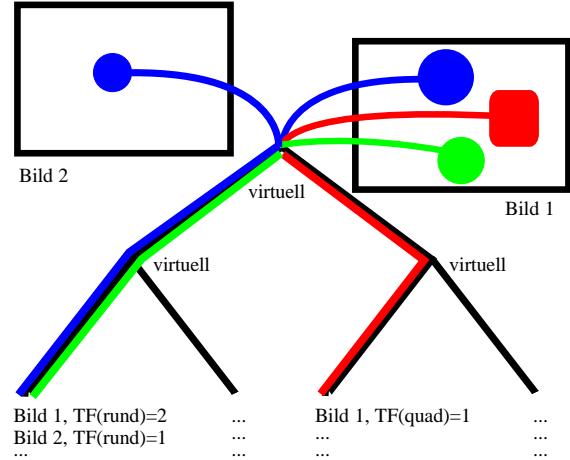


Abbildung 1: Datenbankstruktur mit zwei Ebenen und einem Verzweigungsfaktor von zwei. Merkmale aus verschiedenen Bildern werden quantifiziert und in invertierten Dateien abgelegt. Diese Dateien listen die Häufigkeit und die Bilder in denen das Visual Word gefunden wurde. Dies ermöglicht eine effiziente Suche.

in Abschnitt 2 das Suchverfahren dargelegt, um darauf aufbauend in Abschnitt 3 den VSLAM Algorithmus erklären zu können. Abschnitt 4 befasst sich mit den praktischen Ergebnissen und abschließend wird in 5 ein Fazit gezogen.

2. Scalable Vocabulary Tree

Der hier vorgestellte Ansatz zur visuellen Suche in einer großen Datenbank beruht auf einer Arbeit von David Nistér und Henrik Stewénius. Diese stellten das von ihnen *Scalable Recognition with a Vocabulary Tree* genannte Verfahren [6] bereits 2006 vor.

Es beruht auf der Überlegung, dass bereits sehr effiziente Verfahren zur Textsuche existieren. Beispielsweise beschleunigt ein Index am Ende eines Buches die Suche nach

einem bestimmten Inhalt erheblich. Diese Indexierung wendeten die Autoren auf eine große Bildsammlung an und konnten somit die Suchzeit erheblich verkürzen. Dabei wurde das Konzept der “*Visual Words*” verwendet, die das visuelle Synonym zu Textwörtern darstellen. Das genaue Verfahren wird in den folgenden Abschnitten erklärt.

2.1. Merkmalserkennung

Zur Merkmalserkennung verwendeten die Autoren die *Maximally Stable Extremal Regions* (MSERs) [4]. Diese MSE Regionen werden mittels einer linear affinen Transformation von Ellipsen auf Kreise abgebildet, was den Features eine affine Invarianz verleiht. Anschließend werden diese normalisierten Regionen mit SIFT Feature Deskriptoren [3] beschrieben. Diese Deskriptoren werden durch 128 dimensionale Vektoren repräsentiert.

2.2. Quantifizierung

Um eine kompakte Darstellung und eine schnelle Suche zu ermöglichen, werden die zuvor erkannten Merkmale mittels einer Raumquantisierung eingeordnet. Diese Zuordnung entspricht einer hierarchischen Quantisierung mit einer k -fachen Verzweigung in verschiedene Anhäufungen. Zur besseren Verständlichkeit wird in Abbildung 1 nur eine Datenbankstruktur mit $k = 2$ dargestellt.

Jeder abschließende Knoten stellt eine Menge von Merkmalsvektoren dar, die in ein bestimmtes *Visual Word* (VW) quantifiziert wurden. Jedes VW ist eine einfache Ganzzahl. Die initiale Erstellung des Baumes muss offline mit einer repräsentativen Auswahl von Bildern stattfinden. Dies verhindert eine ungleiche Quantisierung der Feature in den Baum.

2.3. Bildindexierung

Ähnlich der Indexierung von Wörtern in einem Buch müssen nun auch die *Visual Words* indexiert werden. Hierzu wird zuerst mittels der *Inverse Document Frequency* (IDF) die Relevanz eines jeden VW berechnet. Formell ist dies eine Gewichtung der Knoten mit

$$w_i = \ln \frac{N}{N_i} \quad (1)$$

wobei N die Anzahl aller Bilder in der Datenbank ist und N_i die Anzahl der Bilder, die mindestens einen beschreibenden Vektor durch den Knoten i haben. Das bedeutet, dass VWs, die in vielen Bildern vorkommen, eine geringere Gewichtung erhalten.

Nachdem man nun weiß, wie aussagekräftig ein einzelnes VW im Vergleich zum gesamten Vokabular ist, muss noch dessen Häufigkeit im jeweiligen Bild bestimmt werden. Diese Häufigkeit wird *Term-Frequency* (TF) genannt.

Hat man die IDF und TF bestimmt, lässt sich ein Bild als *Visual Word Frequenz Histogramm* beschreiben.

2.4. Inverse Suche

Jeder Knoten im Baum ist mit einer sogenannten invertierten Datei verknüpft. Diese Datei enthält die Bildnummer und die Häufigkeit, mit der dieser Knoten in den einzelnen indizierten Bildern vorkommt. Wie in Abbildung 1 zu sehen ist, erhalten bei der Implementierung von [6] die inneren Knoten (als “virtuell” bezeichnet) keine invertierten Dateien, da ihr Informationsgehalt zu gering ist; sie bestehen nur aus einer Verkettung der abschließenden Knoten. Die Länge der invertierten Datei wird in jedem Knoten gespeichert und spiegelt zugleich den Informationsgehalt des Knotens wider.

Der Anfrage-Vektor q und der Datenbank-Vektor d berechnen sich wie folgt, wobei i für einen Knoten im Vokabular Baum steht.

$$q_i = TF_{q,i} w_i \quad (2)$$

$$d_i = TF_{d,i} w_i \quad (3)$$

$TF_{q,i}$ steht hierbei für die TF des Abfrage-Vektors im VW i . Dies gilt analog für $TF_{d,i}$ und den Datenbank-Vektor.

Möchte man die beiden Vektoren auf ihre Ähnlichkeit überprüfen, müsste man normalerweise die L_2 Distanz berechnen. Dies ist aber sehr ineffizient, da beide Vektoren enorm groß sind und dabei aber nur ein Bruchteil der Elemente überhaupt einen Wert enthält.

Die normalisierte Differenz wird anhand der L_P Norm wie folgt berechnet:

$$\|q - d\|_p^p = \sum_i |q_i - d_i|^p \quad (4)$$

$$= 2 + \sum_{i|q_i \neq 0, d_i \neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p) \quad (5)$$

Für die L_2 Norm ergibt sich:

$$\|q - d\|_2^2 = 2 - 2 \sum_{i|q_i \neq 0, d_i \neq 0} q_i d_i \quad (6)$$

Für eine effiziente Implementierung werden die invertierten Dateien verwendet. Hierbei wird direkt bei jedem Abfrage-Feature, das im Knoten i quantifiziert wird, überprüft, ob das korrespondierende Datenbank-Element $d_i \neq 0$ ist und dann das Produkt $q_i d_i$ aufsummiert.

3. Image Topology based (visual) SLAM

Der von den Autoren Fraundorfer, Engels und Nistér beschriebene Ansatz [2] basiert auf einem topologischen Um-

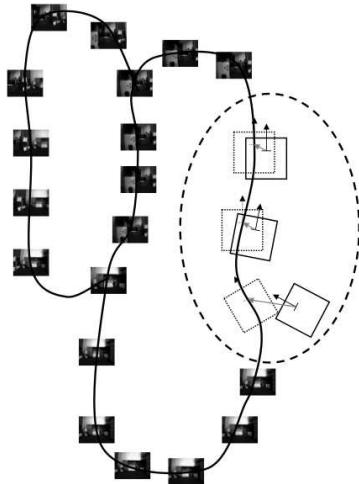


Abbildung 2: Topologische Darstellung einer Karte anhand von verketteten Bildern. Ein Roboter kann basierend auf geometrischen Berechnungen einer bekannten Route folgen.

gebungsmodell, das online erstellt wird. Dieses topologische auf Bildern basierende Modell kann man sich wie in Abbildung 2 vorstellen. Während der Roboter die Welt erkundet, werden laufend Bilder mit einer Datenbank abgeglichen, gegebenenfalls hinzugefügt und darauf basierend eine Verknüpfungskette gepflegt. Dieser Ansatz wurde erst effektiv mit dem Verfahren, das bereits in Abschnitt 2 erklärt wurde. Das visuelle Suchverfahren eignet sich aufgrund seiner kompakten Datenstruktur und seines effizienten Vergleichsalgorithmus für mobile Roboter, die auch in größeren Umgebungen navigieren müssen. Die folgenden Unterabschnitte erklären den Ablauf der Methode und die nötigen Änderungen am Vokabular Baum aus Abschnitt 2.

3.1. Bildakquisition und Datenbanksuche

Aus dem Videostream einer Mono Kamera werden mit bis zu 15 Hz Einzelbilder extrahiert. Für jedes dieser Bilder werden die dazu gehörigen *Visual Words* berechnet. Sobald die beschreibenden VWs zum aktuellen Kamerabild vorliegen, wird in der Datenbank nach ähnlichen Bildern gesucht. Die Suche liefert die n -ähnlichsten Bilder im Merkmalsraum zurück. Nun wird mithilfe von RANSAC [1] und Homographieschätzungen eine geometrische Wertung berechnet. Hierzu werden die gespeicherten Merkmale im Abfrage- und im Datenbankbild verglichen und die Anzahl an Punktübereinstimmungen, die die Homographie ungefähr erfüllen, gezählt. Anhand eines Grenzwertes wird dann entschieden, ob das Bild bereits in der Datenbank vorhanden ist oder noch nicht. Ist das Bild noch nicht vorhanden, wird es inklusive seiner Merkmale gespeichert und ein

Link zum vorherigen Bild im Verknüpfungsgraphen angelegt. Sollte das Bild bereits vorhanden sein, wird die Verknüpfung angelegt. Durch das Anlegen von Verbindungen zu vorherigen Bildern wird eine Schleife in der topologischen Struktur geschlossen.

3.2. Rekonstruktion

Durch den laufenden Abgleich der aktuellen Bilder mit der Datenbank kann immer die aktuelle Position in der topologischen Karte bestimmt werden. Zusätzlich wird aus den relativen Bildinformationen zum aktuellen Bild in der Datenbank anhand von geometrischen Berechnungen die relative Position bestimmt.

Hierfür wird die 5-Punkt Methode nach [5] verwendet, mit der die Essential Matrix E bestimmt werden kann. Aus dieser Matrix lassen sich die Verschiebung und die Rotation zwischen zwei Bildern bestimmen. Die Autoren von [2] gehen davon aus, dass die Kamera fest am Roboter zentriert ist und somit die beiden Koordinatensysteme übereinstimmen. Nun definiert man P_0 als das aktuelle Ursprungskoordinatensystem und P_1 als das Koordinatensystem des gefundenen Bildes in der Datenbank. Dann lässt sich $P_1 = [R|t]$ aus der Essential Matrix E berechnen. Die Rotation R ist hier als 3×3 Matrix und die Verschiebung t als 3×1 Vektor definiert. Der Richtungsvektor vom aktuellen Bild zum Bild in der Datenbank ergibt sich dann wie folgt

$$c = -R^T t \quad (7)$$

3.3. Navigation

Der vorgestellte Navigationsalgorithmus funktioniert nur anhand von Bildern, die als Wegpunkte verwendet werden. Um den kürzesten Pfad zu einem bestimmten Punkt zu finden, zeigt man dem Roboter ein Bild des Ziels. Daraufhin wird aus dem Verknüpfungsgraphen die kürzeste Route anhand von verlinkten Bildern gefunden. Dies bedeutet wiederum, dass der Roboter nur entlang bekannter Pfade fahren kann.

Hat der Roboter eine kürzeste Route zum Ziel gefunden, wird mithilfe der Überlegungen aus Abschnitt 3.2 die Richtung c und die Orientierung R berechnet. Nun beginnt sich der Roboter in Richtung c zu bewegen und stoppt nach einer festgelegten Entfernung d , um sich in die Orientierung R zu drehen. Jetzt findet wieder ein Vergleich des aktuellen Bildes mit der Datenbank statt, ob der erste Wegpunkt bereits erreicht wurde. Sollte der erste Wegpunkt noch nicht erreicht worden sein, wird mit neu berechneter Richtung und Orientierung ein weiteres Teilstück gefahren. Dieser Vorgang wird so lange wiederholt, bis die aktuelle Ansicht bereits dem nächsten Wegpunkt entspricht. Mit diesem Verfahren lassen sich die Wegpunkte mit einer Genauigkeit von

$\pm d$ erreichen und es ist zwingend erforderlich, dass mindestens ein Wegpunkt immer zu sehen ist. Andernfalls bewegt sich der Roboter auf einer zufälligen Route, bis wieder ein Wegpunkt in Sicht ist. Um diesem Fall so gut wie möglich vorzubeugen, beschränken die Autoren die Richtungsänderung auf 10° und die zulässige Drehung für die Orientierung auf 5° .

4. Ergebnisse

Zur Evaluierung der angewendeten Methoden wurden verschiedene Experimente durchgeführt. Hierfür wurde ein nicht holonomer Roboter mit einer Kameraauflösung von 640×480 bei einer Bildrate von 15 Hz eingesetzt. Zur sinnvolleren Nutzung des Sichtfeldes wurde die Kamera in einem 20° Winkel auf den Roboter montiert und mit einem Weitwinkelobjektiv versehen. Für korrekte Berechnungen nach Abschnitt 3.2 musste die Kamera kalibriert werden.

Insgesamt wurden drei verschiedene Szenarien untersucht, die im Folgenden kurz erörtert werden.

4.1. Lokalisierung

Bei diesem Versuch wurde in einem Korridor dieselbe Strecke zweimal abgefahren und dabei Aufnahmen angefertigt. Anschließend wurden diese auf ihre Übereinstimmungen hin untersucht. Bei einem Datenbankbestand von rund 550 Bildern war die minimale geometrische Wertung zwischen den Bildern 9 Treffer. Dies ist immer noch ausreichend, kann jedoch zu Ungenauigkeiten bei der Berechnung der Richtung und Rotation führen.

4.2. Pfad folgen und Heimfinden

Für die Evaluierung wie gut der Roboter einem vorgegebenem Pfad folgt, wurde zunächst eine kurze Strecke abgefahren und dabei die Datenbank mit Wegpunkten angelegt. Anschließend musste diese Strecke autonom wieder abgefahren werden, was sehr gut funktionierte.

Die umgekehrte Richtung des Heimfindens führte zu nicht ganz so guten Ergebnissen, funktionierte jedoch trotzdem und dies auch bei verschiedenen Startpositionen. Hierbei fuhr der Roboter die Strecke jedoch rückwärts ab damit die Kamera immer noch in die Richtung der Bilder für die Wegpunkte zeigt.

4.3. Schleifen schließen

Bei diesem Experiment wurde der Roboter so eingestellt, dass er zufällig durch den Raum wandert und stehen bleibt, sobald er an einem bekannten Punkt ankommt. Dies funktionierte auch wie zu erwarten war. Natürlich kann der Roboter nur dann den gleichen Standort erkennen, wenn er da-

bei auch in dieselbe Richtung wie das ursprüngliche Wegpunktbiß schaut. Nachdem der Roboter an einem bekannten Standpunkt angekommen war, konnte er eine Schleife schließen. Dies führt dazu, dass er von einem beliebigen anderem Punkt in derselben Karte wieder zum Ausgangspunkt zurück findet, ohne die gesamte Strecke noch einmal identisch abfahren zu müssen.

5. Fazit

Theoretisch wird ein effizientes, echtzeitfähiges System zur Navigation in großen unbekannten Umgebungen vorgestellt. Damit dieser visuelle Ansatz funktioniert, muss jedoch eine ausreichende Beleuchtung gewährleistet sein. Die Evaluierungsergebnisse sind leider nicht sonderlich aussagekräftig, da nur ein Bruchteil der möglichen Bildmenge für die Navigation verwendet wurde und diese Menge nicht zwingend auf den SVT als Hilfsmittel zurückgreifen muss.

Interessant wäre ein Ansatz mit einem holonomen Roboter und 360° Kamerabildern für eine direkte Wegführung. Vorstellbar ist auch, dass aufgrund der metrischen Informationen, die für jeden Navigationsschritt berechnet werden, eine Karte angefertigt wird. Dies würde das Mapping auch für andere Anwendungen nutzbar machen. Der Vorteil von 360° Bildern und einer metrischen Karte wäre zusätzlich, dass der Roboter auch erkennen könnte, wenn er an einem bekannten Punkt ist, jedoch in die andere Richtung fährt.

Literatur

- [1] M. A. Fischler und R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, June 1981.
- [2] F. Fraundorfer, C. Engels, und D. Nister. Topological mapping, localization and navigation using image collections. Aus *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, Seiten 3872 –3877, 29 2007-Nov. 2 2007.
- [3] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004.
- [4] J. Matas, O. Chum, U. Martin, und T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. Aus *Proceedings of British Machine Vision Conference*, Ausgabe 1, Seiten 384–393, London, 2002.
- [5] D. Nister. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756 –770, June 2004.
- [6] D. Nister und H. Stewenius. Scalable Recognition with a Vocabulary Tree. Aus *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, Ausgabe 2, Seiten 2161 – 2168, 2006.

Automatische Erstellung von 3D-Objektmodellen mit Hilfe von Tiefeninformationen aus Kamerabildern

Alma Pröbstl
 Technische Universität München
 Lehrstuhl für Medientechnik
 Arcisstr. 21, 80333 München
 alma.probstl@mytum.de

Abstract

Damit sich ein Roboter in einer fremden Umgebung zurechtfindet, soll er Modelle unbekannter Objekte erstellen können. Ein Roboterarm greift hierzu einen Gegenstand und bewegt ihn vor einer Tiefenkamera, welche die Tiefeninformationen aufzeichnet. Daraus wird das 3D-Modell des Objekts erstellt. Um ein möglichst vollständiges Modell zu erhalten, welches auch zunächst verdeckte Ansichten in der Rekonstruktion berücksichtigt, wird ein Algorithmus zur Planung der nächsten Betrachtungsweise des Gegenstandes durch die Kamera und zur Planung eines neuen Griffes vorgestellt. Das Ziel des Algorithmus ist es den Zugewinn an neuer Information zu maximieren.

1. Einleitung

Mit zunehmendem Einsatz und gesteigerter Mobilität von Robotern steigen auch die Anforderungen an ihre Fähigkeiten. Es reicht nicht mehr, dass der Roboter bekannte Objekte erkennt, also ein Objekt einem Modell zuordnet. Der Roboter soll sich zudem auch in unbekannten Umgebungen zurechtfinden, darin neue Objekte erfassen und Objektmodelle anlegen. Langfristiges Ziel ist es, dass die Orientierung eines Roboters im Raum nicht auf den bloßen Sensordaten basiert, sondern ein semantisches Verständnis der Umgebung entwickelt wird.

Ein System, das von unbekannten Objekten eigenständig ein 3D Modell erstellt, wird derzeit von einer Forschungsgruppe an der University of Washington entwickelt. Dieses System [4] wird hier im Folgenden genauer betrachtet. Der Schwerpunkt liegt dabei auf der Entscheidungsfindung, welche Objektansicht als nächstes gewählt werden soll.

Zunächst wird in Abschnitt 2 der Systemaufbau betrachtet. Abschnitt 3 beschäftigt sich mit den Vorteilen, die das Tracking der Roboterhand und des Objekts bietet. Anschlie-

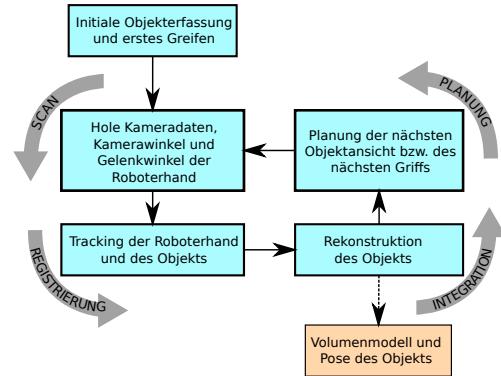


Abbildung 1. Systemaufbau

ßend wird in Abschnitt 4 auf den Ansatz zur Objektrekonstruktion eingegangen. Abschnitt 5 schildert den Algorithmus zur Ansichts- und Griffplanung, den Ansatz zur Qualitätsbeurteilung neuer Ansichten und Griffe, sowie die erzielten Ergebnisse. Abschnitt 6 fasst die Systembeschreibung und den Algorithmus zusammen, bietet einen Vergleich mit anderen Systemen und legt die Zukunftspläne der Autoren dar.

2. Systembeschreibung

Das hier betrachtete System besteht aus einem beweglichen Roboterarm mit Hand, sowie einer auf einem Kipp-Dreh-Gelenk montierten Tiefenkamera. Mit der Roboterhand werden Objekte gegriffen und vor der Kamera bewegt. Ziel ist es, ein 3D Modell des Objekts zu erstellen.

Zur Modellerstellung wird, wie in [6] beschrieben, ein aus vier Schritten bestehender Zyklus durchlaufen. Die vier Schritte sind Scannen, Registrierung (siehe Abschnitt 3), Integration (siehe Abschnitt 4) und Planung (siehe Abschnitt 5). Scannen entspricht dem Einlesen der Daten. Die Registrierung erfolgt in diesem System durch Tracking der Ro-

boterhand und des Objekts. So können mangelhafte Daten zur Kamera- und Objektpositionierung zwischen den Scans ausgeglichen werden. Im Integrationsschritt werden die Daten zu einem einheitlichen Gesamtmodell des Objekts zusammengefügt. Zuletzt wird die nächste Ansicht beziehungsweise der nächste Griff geplant. Abbildung 1 zeigt den Ablauf und stellt den Bezug zu dem hier vorgestellten System her.

3. Objekt Tracking

Obwohl Bewegungen und Gelenkstellungen der Roboterhand aus der Hand-internen Sensorik ermittelt werden können, erfolgt zusätzlich Tracking sowohl der Hand als auch des Objekts. Dies führt zu Robustheit gegenüber Fehlern durch zum Beispiel unpräzise Tiefenmessung und ungenaue Handbewegungen. Das Tracking erfolgt gleichzeitig mit der Objektrekonstruktion. Das Modell der Roboterhand hingegen ist vorher gegeben. Der im betrachteten System verwendete und in [5] beschriebene Ansatz verwendet einen Kalmanfilter und den Iterative Closest Point Algorithmus, welcher die Minimierung der Differenz zweier Punktemengen als Ziel hat.

Tracking bietet mehrere Vorteile. Es kann kostengünstigere Hardware verwendet werden, da man von den registrierten Handbewegungen unabhängiger ist. Der negative Effekt verrauchter Daten verringert sich. Weiterhin können Verdeckungen des Objekts durch die Hand, deren Position durch das Tracking bekannt ist, bei der Rekonstruktion berücksichtigt werden. Das Tracking der Hand erleichtert außerdem die Rekonstruktion sehr symmetrischer und merkmalsarmer Objekte.

4. Rekonstruktion der Objektoberfläche

Aus den Tiefenbildern soll die Oberfläche des Objekts erstellt werden. Die hier verwendete Volumenmethode ist im Detail in [1] und in [4] beschrieben.

Für eine gegebene Menge von Tiefenabbildungen soll die zugehörige Objektoberfläche ermittelt werden. Diese erhält man durch Auflösen der bedingten Wahrscheinlichkeitsdichtefunktion

$$p(d_1, \dots, d_n | S) \quad (1)$$

nach der wahrscheinlichsten Oberfläche unter Verwendung der Maximum-Likelihood Methode. d_1, \dots, d_n stellen dabei die Tiefeninformationen für Aufnahme 1 bis n dar. Die Oberfläche wird mit S bezeichnet. Die resultierende Oberfläche entspricht dem Zero-Level Set der Summe der vorzeichenbehafteten Abstandsfunktionen, wie in [2] bewiesen. Die vorzeichenbehafteten Abstandsfunktionen beschreiben die Distanzen der Messpunkte entlang der Sichtlinie. Sie werden mit dem Kehrwert der Varianz gewichtet.

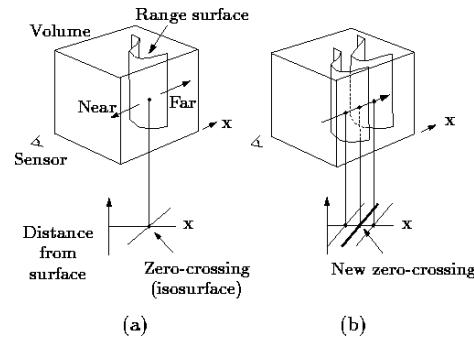


Abbildung 2. (a) Erstellung der vorzeichenbehafteten Abstandsfunktionen aus dem Tiefebild, (b) Interpolation der Zero-crossing Oberfläche aus verschiedenen Messungen; Quelle: [1]

Die vorzeichenbehafteten Abstandsfunktionen sind durch die Messungen gegeben. Aus ihnen extrahiert man das Zero-Level Set, welches die Oberfläche beschreibt. Abbildung 2a zeigt das zu rekonstruierende 3D Objekt und seine Abstandsfunktion in x-Richtung (eindimensional). Der Schnittpunkt mit der x-Achse liefert die Oberflächenkoordinate (hier die x-Koordinate). In unterschiedlichen Messungen unterscheiden sich die Nulldurchgänge und damit die rekonstruierten Oberflächen auf Grund von Messfehlern. Es erfolgt eine Interpolation der resultierenden Oberflächen, wie in Abbildung 2b zu sehen ist.

Die Varianz entlang der Sichtlinie wird zu jedem Oberflächenpunkt gespeichert. Der freie Raum vor dem Objekt entlang der Sichtlinien kann als „leer“ markiert werden, der Raum hinter dem Objekt als „unbekannt“. Lücken im durch das Zero-Level-Set gebildeten Oberflächenmodell werden nun durch hinzufügen der Grenze zwischen „leerem“ und „unbekanntem“ Raum geschlossen. Diesen Punkten werden hohe Varianzen, auf Grund ihrer großen Unsicherheit zugewiesen. Die Position der Hand ist bekannt. Daher können Voxel innerhalb ihres Volumens als „leer“ markiert werden. Die von der Hand verdeckte Oberfläche besitzt eine hohe Unsicherheit.

5. Planung

Viewplanning beschäftigt sich mit der Erstellung einer möglichst kurzen Liste von Ansichten, mit dem die Oberfläche des Objekts bis hin zu einem festgelegten Genauigkeitsgrad rekonstruiert werden kann. Eine neue Ansicht des Objekts kann in dem hier betrachteten System einerseits durch Veränderung der Kameraposition bezüglich des Objekts erzeugt werden, oder durch Veränderung der Stellung der Roboterhand, die das Objekt greift. Für beide Va-

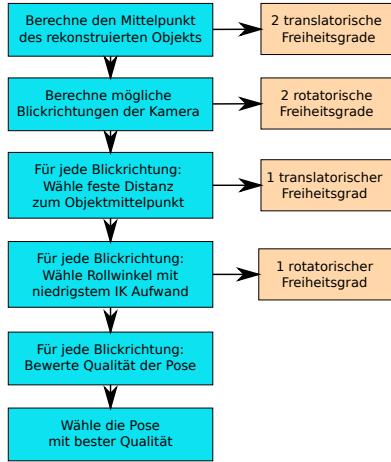


Abbildung 3. Wahl der nächsten Ansicht

riant wird bei der Auswahl der nächsten Ansicht der Informationsgewinn, der mit der neuen Ansicht erreicht wird, ermittelt und damit die möglichen nächsten Kamera-Objekt Positionierungen oder die nächsten Griffe bewertet. Die Bewertung erfolgt für beide Varianten analog.

Die betrachteten Objekte stehen zu Beginn auf einer Oberfläche. Diese Oberfläche wird von den Tiefendaten abgezogen und eine erste Punktemenge des Objekts wird generiert. Dadurch wird das Objekt lokalisiert und ein initialer Griff kann erzeugt werden. Darin liegt ein Unterschied zum klassischen Next-Best-View Problem, bei dem das Objekt auf der Oberfläche stehen bleibt. Ein Vorteil der Modellierung, während das Objekt gegriffen wird, ist es, dass bei genügend häufigem Neugreifen die gesamte Oberfläche erfasst werden kann, während sonst immer der selbe Teil durch die Auflagefläche verdeckt bleiben würde. Allerdings wird die Berechnung zulässiger Kamera Positionen und Orientierungen komplexer, da sie sich nicht auf eine imaginäre Kugel um das Objekt beschränken lassen und auf Grund der Kinematik des Roboters nicht alle Ausrichtungen möglich sind.

5.1. Algorithmus

Zur Wahl der nächsten besten Ansicht müssen die sechs Freiheitsgrade bestimmt werden, die die Kamera Pose festlegen. Der Ablauf des zuständigen Algorithmus ist in Abbildung 3 dargestellt. Zwei der translatorischen Freiheitsgrade werden durch den Mittelpunkt der Oberfläche des betrachteten Objekts beschrieben. Die Oberfläche wird aus dem Volumenmodell der letzten Ansicht extrahiert. Eine Liste der möglichen Blickrichtungen der Kamera, welche zwei der rotatorischen Freiheitsgrade festlegen, wird im nächsten Schritt generiert. Neue Blickrichtungen müssen einen Mindestwinkel zur vorherigen Blickrichtung besitzen. Im Falle

der Griffplanung wird eine Liste zulässiger, stabiler Griffe mit OpenRAVE [3] generiert. OpenRAVE ist eine Software zur Bewegungsplanung mit Anwendung in der Robotik.

Der verbleibende translatorische Freiheitsgrad ist der Abstand zwischen Objekt und Kamera. Er wird auf 0,7 m festgesetzt, unter der Annahme, dass geringere Abstände ein besseres Ergebnis liefern. Der Rollwinkel, als dritter rotatorischer Freiheitsgrad, wird so gewählt, dass der Aufwand in der Roboterbewegung minimal ist.

Die Einträge der Liste der gefundenen Blickrichtungen und den zugehörigen Kameraausrichtungen werden nun bezüglich ihrer Qualität bewertet. Die Bewertung berücksichtigt den erwarteten Informationszugewinn sowie den Aufwand die Position anzufahren. Der Eintrag mit höchster Qualität wird als neue Blickrichtung gewählt.

5.2. Bewertung der Pose Qualität

In Abschnitt 4 ist erklärt worden, dass das rekonstruierte Volumen aus der gewichteten Summe der vorzeichenbehafteten Abstandsfunktionen über n Tiefenbilder besteht. Für jedes Voxel ist zusätzlich ein Gewicht gespeichert. Die Gewichte entsprechen den Kehrwerten der akkumulierten Varianzen der Wahrscheinlichkeitsfunktionen, die den Raumpunkt beschreiben, zum Zeitpunkt t . Für einen neuen Beobachtungspunkt kann nun das zu erwartende Gewicht berechnet werden. Aus altem und neuen Gewicht ergibt sich die Varianz zum nächsten Zeitpunkt ($t+1$). Der Informationsgewinn einer neuen Ansicht bezüglich der aktuellen Rekonstruktion wird durch die Differenz der Entropien zu den zwei Zeitpunkten berechnet. Die Wahrscheinlichkeitsverteilung entlang einer Sichtlinie wird durch die Gaußverteilung approximiert, welche eine Entropie von $\frac{1}{2}\ln(2\pi e\sigma^2)$ besitzt. Der Gewinn ergibt sich zu:

$$G_{t+1} = \frac{1}{2}\ln(2\pi e\sigma_t^2) - \frac{1}{2}\ln(2\pi e\sigma_{t+1}^2). \quad (2)$$

Um die Qualität der neuen Ansicht zu bewerten, werden die Informationsgewinne für alle Sichtlinien für diese Ansicht aufsummiert. Sichtstrahlen die keinen Schnittpunkt mit dem Objekt haben, sich mit der Hand schneiden, zu nahe an der Roboterhand liegen, deren Auftreffwinkel auf das Objekt zu groß ist, so dass der Sensor keine Messung durchführen kann oder die aufgrund eines Stereoeffekts des Sensors keine brauchbare Messung liefern, werden vom Algorithmus nicht in die Berechnung einbezogen.

5.3. Ergebnisse

Abbildung 4 zeigt die Rekonstruktion eines Tetrapacks. Grau markiert sind die beobachteten Oberflächenteile. Je heller der Grauton, desto höher ist das Vertrauen in den Oberflächenteil. Die Grenze zwischen als unbekannt und

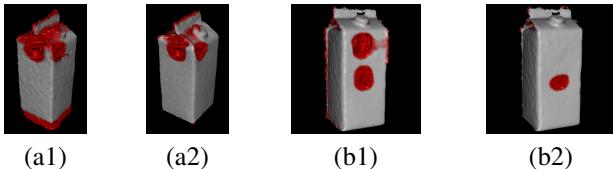


Abbildung 4. Next Best View: (a1) **Modell nach einer Ansicht,** (a2) **Modellansicht nach Ermittlung der nächsten Ansicht. Griffplanung:** (b1) **Modell nach einem Griff,** (b2) **Modell nach zwei Grifffen. Helles Grau = höheres Vertrauen in den Wert. Rot = Grenze zwischen unbekannt und leer. Quelle:** [4]

leer markiertem Raum ist rot gekennzeichnet. Bildteil (a1) und (a2) beziehen sich auf die Planung der nächsten besten Ansicht. Man erkennt, dass sich durch die nächste Betrachtung des Objekts aus veränderter Position die Flächen dunkleren Graues reduziert haben und einige rote, also vorher unbetrachtete, Flächen grau geworden sind. Weiterhin unbekannte Flächen sind zum größten Teil auf die Verdeckung durch die Roboterhand zurückzuführen. Bildteil (b1) und (b2) stellt die Ergebnisse durch Neu-Greifen des Objekts dar. Die Anzahl der rot markierten Flächen ist gesunken. Allerdings verdeckt die Roboterhand wieder eine schon vorher verdeckte Stelle, so dass die Konfidenz in die Rekonstruktion der Objektoberfläche an dieser Stelle gering ist (rot gekennzeichnet).

Kritisch ist der Moment des Neugreifens. Größere ungeplante Bewegungen des Objekts, wie zum Beispiel Umfallen, erzeugen Fehler im Tracking. Auch die Stabilität der Griffe beziehungsweise die Vorhersage der Stabilität ist problematisch.

6. Schluss

Ein System zur automatischen 3D Oberflächenrekonstruktion mit Hilfe von Tiefenkamerabildern, in dem das Objekt in einer Roboterhand gehalten wird, ist vorgestellt worden. Durch das Tracking von Objekt und Roboterhand ist das System robust gegenüber verrauschten Sensordaten. Das Halten des Gegenstandes in der Roboterhand und das erneute Greifen ermöglichen es im Modellierungsprozess Verdeckungen des Objekts in der einen Ansicht in den nächsten Zyklen auszugleichen. Der auf Maximierung des Informationsgewinns basierte Ansatz zur Ansichtsplanung priorisiert Bereiche mit großer Unsicherheit bei der Planung der nächsten Ansicht.

Es gibt weitere Ansätze, die sich mit ähnlichen Problemen beschäftigen. In [7] wird ein System zur 3D Rekonstruktion von unbekannten Objekten vorgestellt. Das betrachtete Objekt wird ebenfalls in Volumendarstellung re-

präsentiert und Voxel-Labeling wird, wenn auch in etwas anderer Weise, verwendet. Der Ansatz unterscheidet sich darin, dass das Objekt modelliert wird ohne es zu greifen. Das Greifen wird als Ziel nach erfolgreicher Modellierung genannt. Weiterhin müssen die nachfolgenden Ansichten überlappen. Das in [8] beschriebene System beschreibt einen ebenfalls informationstheoretischen Ansatz zur Planung der nächsten Ansicht. Die Daten werden dabei auch aus einem vorgelagerten Kalmanfilter bezogen. Allerdings basiert die 3D Objektrekonstruktion auf 2D Daten.

In Zukunft wollen die Autoren des Systems [4] die Griffe nicht mehr extern generieren, sondern einen lernbasierten Ansatz verfolgen. Die generierten Objektmodelle sollen später zum Wiedererkennen und Detektieren dieser Objekte genutzt werden. Ein weiterer Punkt, der verfolgt werden soll, ist die gezielte Minimierung der Griffanzahl. Die Geschwindigkeit der Qualitätsberechnung könnte durch parallele Berechnung des Informationsgewinns pro Pixel unter Verwendung von Shadern optimiert werden.

Literatur

- [1] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.
- [2] B. L. Curless. *New methods for surface reconstruction from range images*. PhD thesis, Stanford, CA, USA, 1998. UMI Order No. GAX98-10106.
- [3] R. Diankov and J. Kuffner. Openrave: A planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, Pittsburgh, PA, July 2008. Online: <http://openrave.programmingvision.com>.
- [4] M. Krainin, B. Curless, and D. Fox. Autonomous generation of complete 3d object models using next best view manipulation planning. Wird erscheinen in ICRA 2011 proceedings, 2011. Online: <http://www.cs.washington.edu/robotics/3d-in-hand/>.
- [5] M. Krainin, P. Henry, X. Ren, and D. Fox. Manipulator and object tracking for in-hand 3d modeling. Wird erscheinen in: IJRR 2011 proceedings, 2011. Online: <http://www.cs.washington.edu/robotics/3d-in-hand/>.
- [6] W. R. Scott, G. Roth, and J.-F. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Comput. Surv.*, 35:64–96, March 2003.
- [7] J. I. Vásquez-Gómez, E. López-Damian, and L. E. Sucar. View planning for 3d object reconstruction. In *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*, IROS'09, pages 4015–4020, Piscataway, NJ, USA, 2009. IEEE Press.
- [8] S. Wenhardt, B. Deutsch, J. Hornegger, H. Niemann, and J. Denzler. An information theoretic approach for next best view planning in 3-d reconstruction. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*, ICPR '06, pages 103–106, Washington, DC, USA, 2006. IEEE Computer Society.

Semantisch bedeutungsvolle Modelle für Visual SLAM

Alexander Schreiber und Werner Maier

Technische Universität München

Lehrstuhl für Medientechnik

{Alexander.Schreiber,Werner.Maier}@tum.de

Zusammenfassung

Im Bereich der Robotik ist der Einsatz von visual Simultaneous Localisation and Mapping Algorithmen (vSLAM) mittlerweile sehr verbreitet. Diese geben dem Entwickler die Möglichkeit während der Laufzeit des Systems eine Karte zu Erstellen, sodass der Roboter sein Umfeld kennen lernt und so auch komplexere Aufgaben erledigen kann. Aufbauend auf diesen Algorithmen wird ein Ansatz entwickelt, der es ermöglicht einen semantischen Bezug der Gegenstände und der Raumstruktur herzustellen, um Aufgaben auf einem höheren kognitiven Niveau erledigen zu können. Hierfür gibt es eine Vielzahl von Ansätzen, wovon zwei im Folgenden näher erläutert und verglichen werden.

Der erste Ansatz benutzt eine monokulare Kamera und erstellt semantische 3D Modelle auf der Grundlage eines bereits existierenden vSLAM Algorithmus. Der zweite Ansatz verwendet eine Stereokamera und Markov Random Fields um einen semantischen Bezug der Oberflächen herzustellen. Beide Ansätze zeigen eine erfolgreiche Herstellung von semantischen Bezügen in Teilen ihrer aufgenommenen Bilddaten.

1. Einleitung

Heutzutage sind Roboter mit einer Vielzahl von Sensoren ausgestattet, die verwendet werden können um den Raum auszumessen und so ihre Umgebung zu erfassen. Hierbei ist es sinnvoll, diese Daten mit Hilfe eines Algorithmus zu einer Karte zusammenzufassen und sich selbst darin zu lokalisieren. In einem weiteren Schritt kann die resultierende Karte verwendet werden um zum Beispiel während der Navigation Hindernisse auszuweichen. Diese Problemstellung wird Simultaneous Localisation and Mapping (SLAM) genannt. Wenn für die Aufnahme der Daten nur eine Kamera als Sensor verwendet wird, spricht man von Visual SLAM (vSLAM). Herkömmliche vSLAM Algorithmen [3] erstellen aus Kameradaten eine Punktwolke, welche darauf hinweist, wo sich Gegenstände oder Wände

befinden. Es werden hierbei im Allgemeinen monokulare Kameras verwendet. Die Algorithmen liefern Schlüsselframes innerhalb des Bildersatzes zurück. Hierbei werden aber keine weiteren Informationen aus dem Bildmaterial erzeugt. Um einen Roboter Aufgaben auf einem höheren kognitiven Niveau abarbeiten zu lassen ist es nötig, Informationen über den Raum in dem er sich befindet, und über die Beziehungen der darin enthaltenen Gegenstände zu sammeln und auszuwerten.

Durch eine derartige Verarbeitung der Daten wird es möglich eine Interaktion zwischen Mensch und Maschine herzustellen. Zum Beispiel könnte ein Roboter ein Glas von einem Tisch im Wohnzimmer zurück zur Küche bringen, wo er dieses in die Spülmaschine deponiert. Durch das gezielte Zuweisen von Labels zu Oberflächen und erkannten Gegenständen findet er dann sowohl das gewünschte Glas, als auch den Tisch und die Spülmaschine.

Der Ansatz von Flint et. al [2] befasst sich mit einer Möglichkeit, den visual SLAM Algorithmus in [3] zu erweitern um neben der Punktfolke die Bedeutung der Gegenstände aus dem gegebenen Bildmaterial zu extrahieren. So wird jeder gefundenen Ebene ein Label zugewiesen.

Olufs und Vincze stellen in [2] eine Methode vor, um aus Bildern von einer Stereokamera einen semantischen Bezug der Oberflächen herauszustellen.

Die vorliegende Ausarbeitung bietet eine Zusammenschau der beiden Ansätze in [2] und [5] und stellt die Ergebnisse dar.

2. Ansätze

Beide Ansätze gehen davon aus, dass die sogenannte *Manhattan World Assumption (MWA)* zutrifft. Diese besagt, dass alle von Menschenhand geschaffenen Gebäude und Gegenstände durch drei orthogonale zueinander gerichtete Achsen beschrieben werden.

Ebenfalls sind beide Ansätze dafür ausgelegt zur Laufzeit der Applikation die Berechnungen durchzuführen und den semantischen Bezug herzustellen.

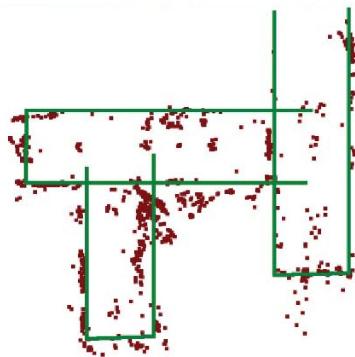


Abbildung 1: Aufnahmen der Kamera und Ergebnis des vSLAM Algorithmus ohne weitere Verbesserungen (Quelle:[2])

2.1. Generierung von semantischen Umgebungsmodellen aus Bildern einer monokularen Kamera

[2] verwendet als Quelle für die Bilddaten eine monokulare Kamera. Der Algorithmus basiert darauf, dass eine vSLAM Implementierung [3] verwendet wird, die die geometrische Struktur der Umgebung und die Kameraposen als Eingabedaten liefert. Diese bestehen nur aus einer Punktwolke, welche schon durch das menschliche Auge keinen direkten Bezug zu Wänden oder Decken zulässt. Ein Beispiel hierfür ist in Abbildung 1 aufgezeigt.

Bei Starten der vSLAM Applikation wird ein Koordinatensystem initialisiert, welches im Folgenden als SLAM-Koordinatensystem bezeichnet wird. Die Anwendung liefert sogenannte Schlüsselframes. Diese werden ungefähr alle 1-3 Sekunden aufgenommen und enthalten jeweils die Kamerapose, die aus einer Rotation und einer Translation besteht, und jene in Abhängigkeit vom Ursprung des SLAM-Koordinatensystems angibt.

In [2] wird darauf hingewiesen, dass durch das Verwenden der Schlüsselframes Frames mit ausgeprägten KantenFrames mit schwer zu identifizierenden Kanten durch die Informationen über die Fluchtpunktrichtungen unterstützen können.

Für die Weiterverarbeitung der Daten ist es wichtig einen Bezug zwischen dem Raumkoordinatensystem, dessen Achsen parallel zu den Kantenrichtungen im Manhattan-Weltmodell gerichtet sind, und dem SLAM-Koordinatensystem herzustellen. Dieser ist durch die Transformation R_ω zwischen beiden Systemen hinreichend definiert.

Um R_ω zu berechnen wird zunächst ein *Canny Edge Detector* [1] verwendet um Kanten in den Schlüsselframes zu finden. Anschließend identifiziert ein Kantenverbindungsalgorithmus [4] gerade Liniensegmente.

Nun werden die Projektionen der drei Fluchtpunkte, die je-

weils in Richtung einer Koordinatenachse des MWA Systems zeigen, im jeweiligen Frame angegeben. Nachfolgend wird eine Fehlerfunktion minimiert um so auf ein optimales R_ω zu schließen. Mit diesem Ergebnis lassen sich SLAM-Koordinaten in Weltkoordinaten darstellen.

Im nächsten Schritt wird die vertikale Richtung aus dem gefundenen R_ω geschätzt. Dies wird erreicht, indem die Achse gewählt wird, in deren Richtung die geringste Kamerabewegung detektiert wurde.

Um z.B. eine Wohnung zu beschreiben ist es notwendig die *Manhattan World Assumption* insoweit zu verallgemeinern, dass man nun davon ausgeht, dass nur ein Boden, sowie eine dazu parallele Decke und vertikal dazwischen gebaute Wände existieren.

Um Flächen eine semantische Bedeutung zuzuweisen ist es notwendig diese Flächen zuvor zu erkennen. Hierzu ist es notwendig Kanten im Bild zu detektieren, sodass diese im weiteren Verlauf zu einer Ebene zusammengefügt werden können. Bei der Erkennung von Kanten werden im Allgemeinen Gradientenfilter verwendet. Diese liefern in Innenräumen ein schlechtes Ergebnis bei schwach erkennbaren Objektkanten Kanten und einem ausgeprägten Gradienten bei Oberflächenstrukturen. Dadurch werden in der Regel Kanten falsch detektiert und würden somit zu fehlerhaften Schätzungen der im Bild befindlichen Ebenen führen.

Flint et al. verwenden zur Lösung dieses Problems eine zweite Suche nach Manhattan Liniensegmenten, also Geraden, die in Richtung einer Achse des Weltkoordinatensystems zeigen. Die Suche wird durch die Informationen über die bekannten Fluchtpunkte beschleunigt und verbessert.

Abschließend werden Hypothesen darüber aufgestellt durch welche gefundenen Linien die Raumstruktur erzeugt wird. Hierzu werden alle gültige Paare von horizontalen Linien gefunden. Diese Raumhypothese besteht aus einer Wand und keinen Ecken. Die Ecken werden hinzugefügt, indem Liniensegmente, welche die Linienpaare schneiden, mit der Raumhypothese in Verbindung gebracht werden.

Die so entstehenden Raumhypothesen B definieren 3D-Modelle bis auf einen Skalierungsfaktor s^* . Um diesen zu bestimmen werden alle SLAM Wegpunkte, die im aktuellen Frame sichtbar sind, mit allen Hypothesen verglichen. Für jeden Wegpunkt wird festgestellt auf welcher Oberfläche innerhalb der Hypothese er sich befindet. Ein temporärer Skalierungsfaktor wird so gewählt, dass die rekonstruierte Oberfläche aus B diesen Wegpunkt enthält. Diese Skalierungsfaktoren werden gespeichert und der Faktor mit der größten Häufigkeit wird für s^* ausgewählt.

Durch diese Berechnungen kann die Raumstruktur von einem Frame zum nächsten übernommen werden und so auch eine semantische Bedeutung zugewiesen werden.

2.2. Generierung von semantischen Umgebungsmodellen aus Stereobildern

[5] verwendet statt einer monokularen Kamera eine Stereokamera. Mit Hilfe dieser können Olufs und Vincze durch Aufnahmen von einer Szene, also im Endeffekt zwei Bildern, die Oberflächen aus $2\frac{1}{2}D$ -Daten rekonstruieren. Auch bei diesem Ansatz ist es wichtig die Orientierung der Kamera festzustellen, was über Entropieminimierung durchgeführt wird. Die MWA-konforme Raumstruktur wird geschätzt, sodass die dominante Struktur zu allen drei orthogonalen Achsen ausgerichtet ist. Ein Nachteil dieser Methode ist, dass räumliche Information über die Voxel, also die kleinste Volumeneinheit, verloren geht. Aus diesem Grund muss hierfür eine Nachverarbeitung stattfinden. Ein Problem, das bei der Schätzung von $2\frac{1}{2}D$ Tiefendaten mit einer Stereokamera auftritt, ist die Unsicherheit beim Finden von Pixelkorrespondenzen. Die Lösung des Problems ist die Verwendung von Ellipsoiden um die Unsicherheit der einzelnen Voxel zu beschreiben. Die Dichte der Ellipsoide werden mit Hilfe zusätzlicher Voxel beschrieben um sie in einem Histogramm darzustellen. Um die Kameraorientierung festzustellen werden drei voneinander unabhängige 1D Histogramme X_x , X_y und X_z für die Voxelkoordinaten X , Y und Z erzeugt. Im nächsten Schritt wird die Shannon-Entropie der Histogramme $\{X_t\}_{t=x,y,z}$ wie folgt berechnet:

$$H(X_t) = - \sum_{i=1}^k p(x_{t,i}) \log_{10} p(x_{t,i}) \quad (1)$$

Hierbei ist k die Anzahl an Linien im Histogramm und $p(x_{t,i})$ der Wert im Histogramm an Stelle i ist.

Somit ist die Manhattan Konfiguration definiert durch

$$\arg \min(H(X_x) + H(X_y) + H(X_z)) \quad (2)$$

also die Konfiguration mit der kleinsten Entropie.

Sobald die Kameraorientierung bekannt ist, können Hypothesen über die Ebenen erstellt werden. [5] unterteilt dies in drei Schritte.

1. Generieren von individuellen 2D Histogrammen von allen möglichen Kombinationen von X, Y und Z
2. Extrahieren der Liniensegmente aus den 2D Histogrammen
3. Zusammenfassen der Segmente zu Ebenen

Das Extrahieren der Liniensegmente wird nur auf die vertikal und horizontal ausgerichteten Geraden angewendet.

Auch bei diesem Ansatz wird der *Canny Edge Detector* verwendet um Kanten zu detektieren. Der Algorithmus wird in dieser Implementierung verwendet um den Kanten

Labels zuzuweisen. So stellt dieser fest, ob ein Pixel *eine Kante*, *vielleicht eine Kante* oder *keine Kante* ist. Durch weitere Aufteilung des Bildes und weitere Berechnungen werden den Pixeln mit dem Label *vielleicht eine Kante* das Label *eine Kante* zugewiesen, sofern ein Pixel mit diesem Label in der Nähe auftreten sollte. Folglich werden alle zusammengehörigen Pixel mit dem Label *eine Kante* zu Liniensegmenten zusammengefasst.

Diese Liniensegmente lassen sich nun zusammenfassen, damit Hypothesen über Ebenen entstehen. Eine solche Ebene wird geschätzt zwischen einem Liniensegment und einem anderen, welches die Normale dazu bildet. Die besagten Linienelemente sind in Richtung der Achsen ausgerichtet.

Die Ebenenhypothesen werden im nächsten Schritt dafür verwendet um die Voxelmenge vorab zu segmentieren, sodass ihr gewisse Ebenen und deren Orientierung zugewiesen werden. Jedes Voxel wird hierbei nur einer Ebene zugewiesen. Sollten Hypothesen zu wenige Voxels zugewiesen bekommen, werden sie aus dem Speicher gelöscht. Um eine Segmentierung des gesamten Bildes vorzunehmen werden die Pixel mit gleicher oder sehr ähnlicher Farbe zu Superpixel zusammengefasst. Es wird in [4] davon ausgegangen, dass Objektgrenzen auch durch die Grenzen der Superpixel dargestellt werden.

Um die Pixel in einen globalen Kontext zu bringen wird das segmentierte Bild als probabilistischer Graph betrachtet und ein *Markov Random Field* (MRF) als Modell verwendet.

Hierbei wird davon ausgegangen, dass eine Menge an Beobachtungen P , in diesem Fall die Voxel, bzw. Superpixel, und eine Menge an Labels L , in diesem Fall die Ebenenhypothesen, gegeben sind. Das Ziel ist es jeder Beobachtung $p \in P$ ein Label $f_p \in L$ zuzuweisen, sodass es die Benennungsfunktion $E(f)$ minimiert.

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{pq \in N} V_{pq}(f_p, f_q) \quad (3)$$

Hierbei stellt $D_p(f_p)$ einen Datenkosten-Term und der zweite Summand die Kosten für die Glattheit der Labels dar.

3. Ergebnisse

Abbildung 2 zeigt, dass [2] die Zuweisung von Labels innerhalb eines Flures korrekt vornimmt. So ist deutlich eine Abgrenzung des Bodens von den Wänden, sowie der Decke zu erkennen. Es ist auffällig, dass die Türen zusammen mit den Wänden verbunden sind. Innerhalb des Bildes wird jeder Abschnitt einem Label zugewiesen, sodass Bereiche, die unter Umständen nicht zu der Oberfläche gehören, nicht weiter betrachtet werden.

Olufs und Vincz [5] zeigen, wie in Abbildung 3 dargestellt,



Abbildung 2: Zusammenfassung der Resultate aus [2]
Dargestellt wird hierbei der Versuch, Oberflächen in einem Büro Labels zuzuordnen.

dass ihr Algorithmus auch kleinere Oberflächen erkennen kann. Auch ist, wie schon bei [2], der Boden korrekt erkannt. Allerdings bleiben viele Bereiche des Bildes ohne Zuweisung von Labels. Hierbei weisen Olufs und Vincze darauf hin, dass ihr Ansatz Schwächen aufweist, sobald die Tiefendaten aus dem Stereobild nicht ausreichend vorhanden sind.

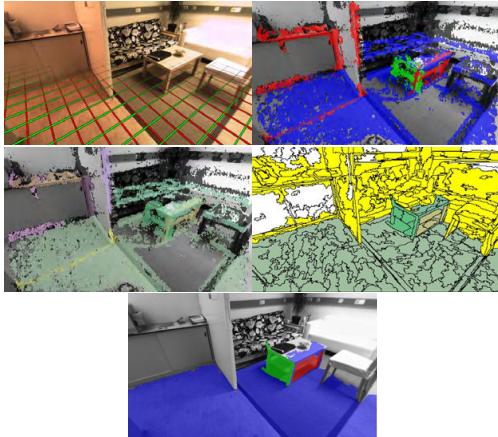


Abbildung 3: Ein Beispiel der Ergebnisse aus [5]
Schätzung der Kameraorientierung. Ausrichtung zu einer Achse und Zuweisen von vorsegmentierten Pixeln. Erste Zuweisung von Farben zu den vorsegmentierten Pixeln. Erste Zuweisung von Farben zu den finalen MRF Labels mit Hilfe der Superpixel. Hauptachsenzuweisung der finalen MRF Labels.

4. Fazit

Beide Algorithmen haben gezeigt, dass eine Online-Verarbeitung von semantischen Daten möglich ist und bieten somit die Möglichkeit Robotern mehr Intelligenz zu

vermitteln um Aufgaben auf einem höheren kognitiven Niveau zu lösen. Es bleibt allerdings fraglich ob die aktuellen Ansätze ausreichend sind um einen ausreichenden semantischen Bezug eines gesamten Raumes herzustellen, da sowohl [2] als auch [5] einige Schwächen aufzeigen.

Beide Ansätze beginnen ihren Algorithmus mit der Feststellung der Kameraorientierung. [2] verwendet hierbei die schon bekannten vSLAM Positionen und Transformationen, [5] die minimale Shannon Entropie. Beide versuchen dann Linien zu finden, die in die Richtung der durch die *Manhattan World Assumption* festgelegte Koordinatensystem-Achsen, orientiert sind. Sobald diese gefunden werden, werden Hypothesen über die Ebenen, bzw. die Oberflächen im Bild erstellt und dann anhand eines eigenen Testalgorithmus überprüft und aussortiert.

Der letzte Schritt ist bei beiden die Zuweisung von Labels zu diversen Oberflächen. [2] verwendet hierzu die Fluchtpunkte und die gefundenen Geraden, sowie eine Schätzung mehrerer 3D-Modelle um schließlich das korrekte zu finden. [5] hingegen bedient sich der *Markov Random Fields* und einer komplexen Histogrammauswertung.

5. Ausblick

In dieser Arbeit wurden zwei Ansätze dargestellt, die aus rudimentären geometrischen Modellen semantische Strukturen extrahieren. Durch diese wird es möglich einem Roboter zum Beispiel beizubringen, wo sich Wände oder Decken im Raum befinden. Auch ist es denkbar, dass er Stühle oder andere Möbel erkennt und so eine Aufgabe wie "Bringe Bierflasche von Kühlschrank zu Tisch" erledigen kann.

Literatur

- [1] J. Canny. A Computational Approach to Edge Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8(6):679–698, Nov. 1986.
- [2] A. Flint, C. Mei, I. Reid, and D. Murray. Growing semantically meaningful models for visual SLAM. In *Proc. of Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 467–474, June 2010.
- [3] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234, Nov. 2007.
- [4] J. Kosecka and W. Zhang. Video Compass. In *Proc. of the 7th European Conference on Computer Vision-Part IV, ECCV '02*, pages 476–490, London, UK, UK, 2002. Springer-Verlag.
- [5] S. Olufs and M. Vincze. Robust Single View Room Structure Segmentation in Manhattan-like Environments from Stereo Vision. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5315–5322, Shanghai, China, 2011.