Plenoptic Models in Robot Vision

Joachim Denzler, Benno Heigl, Matthias Zobel, Heinrich Niemann

Plenoptic models, representatives are the lightfield or the lumigraph, have been successfully applied in computer vision and computer graphics in the past five years. The key idea is to model objects and scenes using images and some extra information like camera parameters or coarse geometry. The model differs from CAD–models in the photorealism that can be achieved and is thus superior in applications where the realism of the model is of importance. From the point of view of learning based robot vision these models have the additional advantage that they can be acquired fully automatically from image sequences.

The paper shows how such models can be applied in robot vision. Starting with the theoretical principles, the automatic generation of plenoptic models is discussed. An new method is introduced for direct rendering from arbitrarily taken real views. The adaptive use of geometric information makes it possible to scale the model accuracy with respect to available computation time.

In two typical applications from robot vision the benefits of this kind of model is demonstrated. The experimental results proof our claim that plenoptic models are useful in robot vision.

1 Introduction

Models of objects and scenes and automatic model building is one important aspect in robot vision. The main criteria for good models are uniqueness and stability. A model is applied by comparing the information stored within in the model with the recorded image data. This general matching procedure returns the information that best matches the model with the image data, for example the 3–D position and rotation angles of an object in 3–D, or the class number, for which the a posteriori probability of an object given the image data is maximized. Since every kind of model shall serve as an abstraction of the object or scene in terms of primitives, the problem is, which kind of primitives the model shall consists of, and how difficult it is to reliably extract those primitives from image data. Primitives might be matrices about pixel brightness, segmentation objects like lines, curves, corners, or geometric representations to hold knowledge about 2D and 3D shapes.

Models based on CAD descriptions (wire frames, or surface boundary representations like polygons, B–splines, etc.) are advantageous if one has to deal with man made objects. A problem is how to automatically build such a model from sample views, if a CAD model is not readily available. Statistical models on the other hand are able to model statistical properties of primitives. Also model building is straight forward using techniques from parameter estimation. The problem is that in general a huge number of training images is necessary to robustly estimate the parameters of the statistical model, or in other words, to find those parameters, that represent the object or scene adequately.

In this article, we propose a new kind of model for robot vision, the so called *plenoptic model*, a model that can be acquired fully automatically. This is regarded in this article as learning. The key idea is to collect a sequence of images from objects or scenes using a hand–held camera. These images together with some extra information, which is computed automatically during model building, serve as a model of the object or scene. Using this model photo–realistic views from objects and scenes can be created (rendered) synthetically, even if the view has not been observed before.

In the next section we give a short literature overview on plenoptic models. In Section 3 we describe how we calibrate an image sequence, which is preliminary for model building. Techniques for rendering from the plenoptic model are presented in Section 4. All methods for rendering from the plenoptic model up to now make assumptions that are not valid when taking image sequences with a hand–held camera. Thus, we developed a new, efficient method for direct rendering from real views with local geometry (Section 5). In Section 6 two applications of plenoptic models are presented together with results that show the benefits and problems of the method. The paper concludes with references to other applications of plenoptic models and an outlook to future work.

2 Plenoptic Modeling

The *plenoptic function* [1] describes everything that can be seen within a scene as a function of a view point and viewing direction returning a color value. If an image is taken at position $\boldsymbol{t}$, a whole bundle of viewing rays is recorded (see Figure 1). Together with its color value, each ray corresponds to one sample of the plenoptic function that is given here as a three–valued function of the pixel position $\boldsymbol{q}$ and a projection matrix $\boldsymbol{P}$. The function values are the three components of the observed color represented by RGB–values.

By taking lots of images from different view points we get a more or less dense sampling of the plenoptic function.

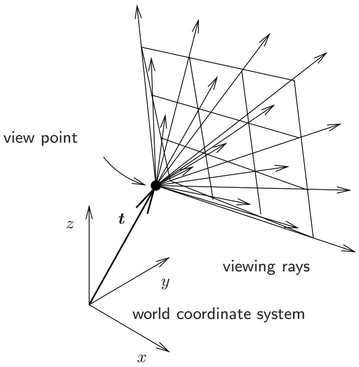

Figure 1: The plenoptic function measures the color that can be observed at the point $\boldsymbol{t}$ in a direction that is specified by a given pixel position and the camera parameters. A whole image corresponds to a bundle of viewing rays intersecting at the projection center of the camera. Each viewing ray is one sample of the plenoptic function.

Having the complete plenoptic function it would be easy to render new virtual views from the scene by just composing the new image evaluating the plenoptic function for each pixel.

In practice, only a finite number of scene views can be recorded and therefore we never will be able to determine the complete plenoptic function. Assuming a fully transparent medium the observed intensity does not change if the viewing position is changed along a viewing ray. The observable color value depends on the selected viewing ray, not on the point of observation and therefore one dimension can be reduced.

In [13, 6] the *light–field* approach has been introduced to render views efficiently if the viewing rays are sampled in a fixed regular structure. Two planes in space are fixed and on each plane a discrete grid is defined. All viewing rays passing one grid point on each of the two planes can be parameterized.

To build this regular structure a robot arm moves a camera to grid positions in one plane viewing an object. Because of its regular structure, the two–plane parameterization is not very flexible and does not fit to an image sequence being recorded with a hand–held camera. The only advantage of the regular structure is that it is very suitable for efficient rendering using texture mapping hardware. In [6] a way has been shown to interpolate the light field structure from arbitrary posed camera views. This method entails a loss of accuracy solely for creating this regular data structure called *lumigraph*

In [14] a plenoptic scene model is built from two panoramic views taken at distinct viewing positions. Each panoramic view is composed of many camera images that are recorded in many directions with a coinciding projection center. Although having many viewing rays passing the two viewing positions, the overall–sampling of viewing rays is not very dense.

Instead of modeling a viewing ray originating from a view point in a given direction, it can be seen as a ray that ends at a given scene point in a given direction. The dependency of a surface point on the viewing direction is covered in the so–called *bidirectional reflectance distribution function* (BRDF) [5, p. 663] that models the relation between incoming and outgoing radiance. For the appearance of a point it does not matter how its color value depends on the incoming radiance and it consequently can be modeled by the color values of all these viewing rays. Therefore samples of the plenoptic scene model also can be represented by modeling the surface geometry of the scene and its appearance in different directions. This approach has been used in [3, 20]. In [3] the geometry of the scene is modeled by geometrical primitives and their appearance is represented by a set of texture images with associated viewing directions. In [20], the geometry is modeled by surface points and so–called *lumispheres* being associated to each of these points. This modeling has the advantage to be very compact, but its disadvantage is that the geometry of the surface must be known exactly. To get the exact geometry in [3] the user must interact to reconstruct geometrical primitives like plane patches or cylinders. In [20] a range–scanner is applied to ensure geometrical accuracy.

For our approach, we decided to represent recorded plenoptic samples as the whole set of recorded images together with retrieved camera parameters. This strategy simplifies the process of building up the plenoptic model but it claims increased demands on the rendering method.

3 Calibration

For plenoptic scene modeling the movement and the projection properties of the recording camera must be known. We compute this information from image data alone with methods we have described in detail in [8]. Here only a short overview is given.

In literature lots of approaches exist for determining camera pose and projection properties from multiple images. We call this task *calibration*. An extensive overview over most techniques is given in [7]. For the case of calibrating lots of images and lots of feature points the so–called *factorization methods* solve the problem simultaneously within a single step for all the data [15, 18]. Mainly because of this advantage we decided for this class of techniques. Before a factorization method is applicable feature point correspondences must be extracted from the input images.

The goal of tracking is to determine the 2-D locations of one common 3-D point in many projections. In our experiments we apply the differential tracking method described in [17] for extracting these 2-D locations from an image sequence resulting in so–called *trails*. This method approximates the image function by the linear term of a Taylor series and minimizes a residual function defined over a feature window by setting the derivatives to zero resulting in a linear equation system. The algorithm is designed such that either a pure displacement or additionally an affine transformation of a feature window can be considered. We apply the estimation of a pure displacement for tracking a feature window from frame to frame. Affine distortions are considered to re–adjust the location of the feature window by comparing the currently tracked window with the feature window of the first occurrence. With this technique error accumulation during tracking can be avoided.

As mentioned above, the factorization methods are capable of solving the calibration problem for many views simultaneously. But there exists the drawback that all considered feature points must be visible and detectable in all frames. As this constraint is not fulfilled in our case we only can start with a partial sequence, which is completely covered in many trails. In [8] we have developed an algorithm that automatically finds a partial image sequence from a given set of trails, that is as long as possible for a given minimum number of completely visible trails.

The basis of our calibration are the factorization methods of [15] and [18]. Where in [15] a weak–perspective camera model is assumed, in [18] the approach has been extended to the perspective projection model. The first approach only needs the pure 2-D point information but introduces a bias of the solution because of the approximative weak–perspective projection model. The second approach additionally to the 2-D points needs information about the depths, called *projective depths*, but if this information is available at least approximatively the result is better than for the first approach.

We combine the two approaches as follows: first we apply the weak–perspective factorization method, we convert the weak–perspective projection matrices into perspective ones, and add a non–linear optimization. From this result we determine the projective depths and take them as input for the perspective factorization method followed by a self–calibration step and a non–linear optimization.

4 Visualization by Interpolation and Extrapolation

We have seen several examples for the acquisition and representation of plenoptic scene models. In this section methods are discussed being capable of rendering new views from these models.

4.1 Interpolation with known depth

Each viewing ray can be interpreted as the color value of surface point $\boldsymbol{w}$ seen from a given direction. The color value of a virtual viewing ray passing $\boldsymbol{w}$ can be determined by interpolating between the neighboring rays. This method is used by the techniques [3, 20]. When generating such plenoptic scene models, the appearance of one scene point $\boldsymbol{w}$ from many different viewing directions must be determined from recorded images. The reliability of the object–centered interpolation highly depends on the accuracy of the surface geometry. Because of this reason, these systems do not reconstruct the geometry automatically from the image data but require user–interaction or additional hardware to retrieve highly accurate depth information.

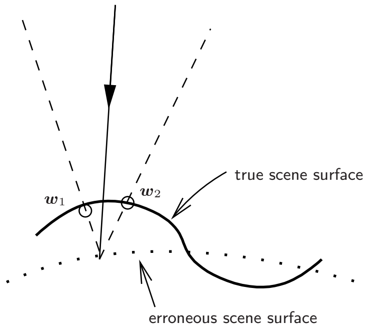

Figure 2: Interpolation with depth uncertainty. A viewing ray (solid line) is interpolated from two recorded viewing rays (dashed lines) that have been selected assuming an erroneous scene surface.

4.2 Interpolation with depth uncertainties

In our case depth information is reconstructed from the input images by interpolating the 3-D scene points being a side–product of the calibration procedure. This approach is not very accurate and therefore the object–centered interpolation cannot be applied here. Nevertheless photo–realistic rendering is possible even when the geometry is just known approximately. The theoretical proof of this fact has been derived in [2] for the light field parameterization.

Figure 2 shows how artefacts are caused. When interpolating the color values from two different viewing rays assuming an erroneous surface point, then distinct parts of the scene surface are overlayed causing a “doubled” appearance.

To reduce this effect, in [6] a method is developed considering an approximate geometry model when using the light field data structure. Related to this approach is the method [16] that assumes to have two recorded views and that is able to render virtual views from viewpoints lying on the connecting line of the original viewpoints.

4.3 Extrapolating views

In the following we discuss methods for extrapolating views meaning that the virtual viewing position does not need to lie in the range of the viewing positions occurring during acquisition.

It is assumed that besides one or more recorded views a model of the geometry is available. The idea is to map the original view onto the surface of the scene and to view this mapped texture from a new, virtual viewpoint. To do this, it is assumed that a surface point appears identical from all viewing directions, therefore the surface is assumed to be Lambertian. In [14, chapter 3] the theoretical background of this technique is discussed.

The warping methods have the disadvantage that no view–dependent changes are modeled and therefore effects like specularities are not modeled accurately. Moreover it is assumed that the geometry of the surface is perfectly known. As already mentioned, the surface geometry that can be reconstructed automatically from the image sequence has limited accuracy. When using it for view–extrapolation, conspicuous distortions appear.

5 Direct Rendering from Real Views and Local Geometry

In the last section we have seen different methods for rendering views from a given set of plenoptic samples. For our application, none of these approaches fits perfectly. The Lumigraph approach [6] has the advantage of allowing to consider depth information for view interpolation. But this approach needs a 3-D model of an object being consistent to all input views, it is bound to the regular light–field structure, and it is not designed for extrapolating views. The approach of view–dependent texture mapping [3] is able to extrapolate views, but it needs an exact geometrical model for reliable results and it is not able to interpolate suitably between densely spaced views.

We see a gap between these approaches filled by the method described in the following. The main requirements that are met in our approach are:

- We render directly from the originally recorded images without creating a special light–field structure.
- We use local depth maps instead of a global consistent geometry model, since it is not trivial to fuse local depth information to build up a reliable global model.
- Our approach is scalable. This means that there exists a possibility to adjust the quality of rendered results in favor of computational speed.

5.1 Mapping via triangles

For the following methods it is substantial to map an image onto a 3-D triangle and vice versa. The triangle is built by the three points $\boldsymbol{x}_i$, $1 \le i \le 3$. Each point $\boldsymbol{w}$ within the triangle can be represented by

$$\boldsymbol{w} = \begin{pmatrix} \boldsymbol{x}_2 - \boldsymbol{x}_1 & \boldsymbol{x}_3 - \boldsymbol{x}_1 & \boldsymbol{x}_1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix}. \qquad (1)$$

The point $\boldsymbol{w}$ is perspectively projected into a given camera by $\underline{\boldsymbol{q}} \sim \boldsymbol{P}\underline{\boldsymbol{w}}$ where $\underline{\boldsymbol{q}}$ is the 2-D point in homogeneous coordinates, $\underline{\boldsymbol{w}}$ is the 3-D point in homogeneous coordinates, the symbol $\sim$ means the equality up to scale and $\boldsymbol{P} \in \mathbb{R}^3 \times \mathbb{R}^4$ is a projection matrix. Let $\boldsymbol{P}$ be represented as the concatenation of $\boldsymbol{M} \in \mathbb{R}^3 \times \mathbb{R}^3$ and $\boldsymbol{m} \in \mathbb{R}^3$: $\boldsymbol{P} = (\boldsymbol{M}|\boldsymbol{m})$. Together with equation (1) this yields

$$\underline{\boldsymbol{q}} \sim \boldsymbol{M} \underbrace{\begin{pmatrix} \boldsymbol{x}_2 - \boldsymbol{x}_1 & \boldsymbol{x}_3 - \boldsymbol{x}_1 & \boldsymbol{x}_1 + \boldsymbol{M}^{-1}\boldsymbol{m} \end{pmatrix}}_{\boldsymbol{H}} \begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix}. \qquad (2)$$

Therefore each mapping between a local plane coordinate system and a camera image can be described by the 3×3 homography matrix $\boldsymbol{H}$. Vice versa by multiplying $\boldsymbol{H}^{-1}$ by a homogeneous image vector, one gets the corresponding point in the coordinate system of the triangle.

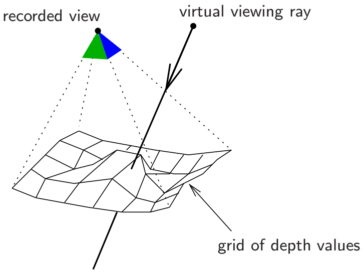

Figure 3: Intersection of the viewing ray with the depth map as search between the planes of minimum and maximum depth.

We can extend this mapping procedure to re–project the image of one camera (marked by index 1) onto the triangle followed by a projection into another camera (marked by index 2): $\underline{\boldsymbol{q}}_2 = \boldsymbol{H}_2{\boldsymbol{H}_1}^{-1}\underline{\boldsymbol{q}}_1$. The matrix $\boldsymbol{H}_2{\boldsymbol{H}_1}^{-1}$ describes the projective mapping from one camera to another via a given triangle. As projective mapping can be performed by graphics hardware, this step can be done in real–time.

5.2 Combined interpolation and extrapolation

In the following a technique is described being capable of combining the advantages of view interpolation with the ability of extrapolating views without the necessity of distinguishing explicitly between these two aims. This technique is based on the results given in earlier publications. In [12] an approach is shown that assumes a single plane as approximation of scene geometry. This approach has been extended in [11] to be capable of interpolating between views with associated depth maps. Here we describe the extension for extrapolation. The complete algorithm is described in detail in [8].

Suppose having a single view together with the according depth map as visualized in Figure 3. For reconstructing the color value of a given viewing ray, the intersection point with the scene surface is needed. Unfortunately this point cannot be found by a simple look–up. It rather must be found by searching. In [19] an efficient way for finding this hit point is described.

We choose a regular triangulation of grid positions in the virtual destination image as shown in Figure 4. For each grid position it is tried to determine 3-D points by applying the search method mentioned above to each recorded image. Because of occlusion and errors in depth maps this method in general leads to multiple hypotheses for the according 3-D points. To judge the different hypotheses, for each grid position j weight factors ξ_{jk} are determined for all 3-D points $\boldsymbol{w}_{jk}$ that have been recovered in the views k: $\xi_{jk} = (\pi - |\alpha|)^2$, where α is the angle between the virtual viewing ray corresponding to the 3-D point $\boldsymbol{w}_{jk}$ and the particular source ray.

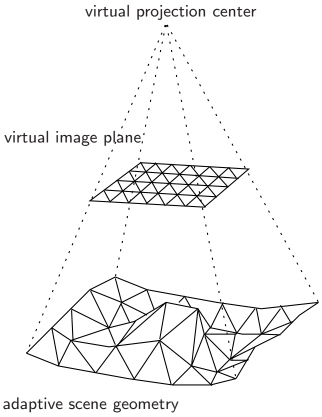

Figure 4: Adaptive scene geometry. For a regular image grid the adaptive scene geometry is determined by intersecting viewing rays of grid points with reconstructed depth maps.

Each triangle of the virtual view is drawn by overlapping up to N_{v} triangles from different recorded views. In our experiments we have chosen $N_{\text{v}} = 5$. For each triangle all contributing source views are determined. If more than N_{v} views could contribute to a particular triangle, those N_{v} views are selected that provide the highest weights. They are mapped from the recorded views into the virtual view via the 3-D plane defined by the particular triple of 3-D points with the method described above. These triples in general are not identical for different views, because it was not ensured that the vectors $\boldsymbol{w}_{jk}$ are equal for different indices k.

The contributing views are overlayed by weighting and adding the triangles that are mapped from the original views into the virtual view. The weights of a contributing triangle at each triangle corner is determined by the weight ξ_{jk} divided by the sum of weights of all contributing views. This scaling ensures that the total sum of weights is 1.

The scalability of the approach is obtained by adjusting the resolution of the initial triangulation. When increasing the number of grid points, the number of reconstructed 3-D points also is increased and therefore the surface geometry is sampled more exactly at the expense of additional computational cost. If we want to reduce computational complexity, the number of grid points can be decreased and therefore the geometrical approximation is more coarse in favor of computational speed.

If the virtual view is outside the viewing range of the recording camera the views are extrapolated. Those views are overlayed that are next to the virtual viewing position. Having errors in the depth maps, geometrical distortions become more obvious the larger the distance to the next recording camera gets.

This approach enables the rendering of virtual views from an arbitrary image sequence recorded in advance. Applying the calibration procedure described in Section 3, no additional information about the scene or recording conditions is necessary.

6 Applications

Two examples are presented in the following. In the first example, object tracking, it is shown how plenoptic models serve as object model directly in an robot vision task. In the second example the whole scene is modeled. The model is taken for localizing a mobile robot using a particle filter approach.

6.1 Model Based Object Tracking

In template based object tracking the prediction of the appearance of the object in the next image is the most important aspect for successfully tracking a moving object. Without a suited update mechanism, varying appearance of the object due to rotation, is one of the main reasons for a failure during tracking.

Plenoptic models are a perfect model for the mentioned update mechanism of the template. If tracking is done in 3–D and also the pose of the object is estimated, the appearance of the object in the next image can be predicted.

Different experiments have been performed for sequentially estimating the pose of a moving object using a particle filter approach. In particle filters, each particle can be interpreted as a hypotheses about the true position and pose of the object. The weight of the particle is proportional to the likelihood of that hypotheses. During the reweighting (or updating) of the particles, the weight (or probability) of each particle has been computed by comparing the acquired image with the image that should be seen if the hypotheses would be true. The imaginary image is rendered from the plenoptic model.

To retrieve ground truth data we simulated a movement of a toy elk by moving a camera over a predefined path on a hemisphere around the elk. In a training step a plenoptic model of the elk has been reconstructed. During tracking for a given estimate of the pose of the object a synthetic image is rendered using the plenoptic model. Comparing the real image with the synthetic one result in a rating of estimated pose.

In Figure 5 the tracking results are shown. By visually comparing the real and the rendered images the reader will observe that the estimated pose of the elk is quite accurate, although the brightness between real and rendered images varies. The mean estimation error in the pose for different movement paths on the hemisphere was between 2.4 degree and 3.4 degree. The computation time for 100 particles in the particle filter approach is 14 secs per frame on a Pentium III/800 MHz, which is far from being frame-rate. One problem is the huge numbers of particle that are usually necessary for reliable state estimation.

6.2 Vision Based Ego–Localization

Plenoptic models might also serve as scene models. In the area of robot vision and ego–localization they compete with classical models, like CAD–models of the environment. The advantage of plenoptic models is obviously the photo–realism.

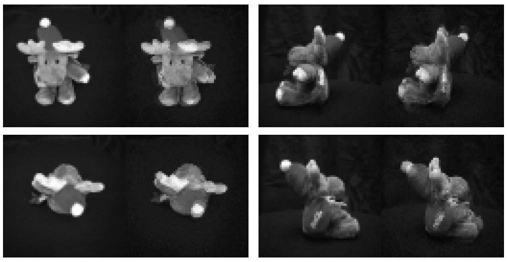

Figure 5: Comparison between the original image (left) of the tracked elk and the rendered image (right) according to the estimated 3–D position for frame numbers 0, 17, 35, 53.

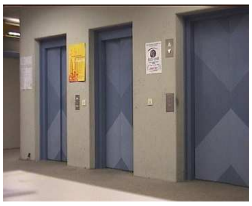

Figure 6: The robot's scene environment. In the test sequence, the robot starts from the left and moves towards the left elevator button.

For ego–localization a plenoptic model of the scene shown in Figure 6 has been reconstructed automatically. During reconstruction knowledge about the scene geometry has been used. By means of that scene model the task of the robot was to move from an arbitrary, initially unknown position in the scene to the elevator buttons between the left and the middle elevator doors. The robot localized its position iteratively based on the image data only and taking into account the odometric information about its last movement. Since the odometry is known as a quite accurate cue for short but being inaccurate during longer movements iterative ego–localization has been performed again using a particle filter approach similar to [4].

In Figure 7 results of the ego–localization is visualized for three different time steps. In the left image the trajectory of the robot and the estimated hypotheses of the robot's position is shown (starting with a uniform distribution in the beginning). The middle image shows the image taken by the camera, the right one the rendered image based on the best hypotheses. Using 195 particle every 2 seconds a position estimate is returned. More details can be found in [10].

7 Conclusion

In this paper we proposed a new type of model for robot vision applications, so called plenoptic models. The advantage of the approach is that for any object or scene, that can be recorded with a hand–held camera, such a model can be reconstructed and new, previously unseen viewpoints can be selected and respective images can be rendered from the model.

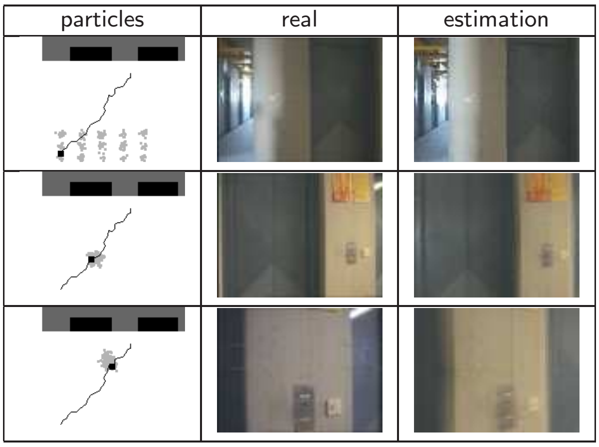

Figure 7: An experiment for testing the localization capability of our approach when using the robot's odometry. Left: schematic top view of the area in front of the wall (gray area) with the elevator doors (black rectangles). Middle: real camera view. Right: rendered image at the estimated most likely position or the robot.

The experimental results of the paper have shown, that this kind of model is useful as object model in object tracking, and as scene model in vision based ego–localization. Besides these two examples, plenoptic models can also be used in model learning itself. Thanks to the photorealism, synthetically generated images can be used as training images. For example, a statistical model for object recognition can be trained [9], or a Markov decision processes for active object recognition [8]. The benefits consist of a reduced effort for image acquisition, which might be a time consuming task due to hardware limitations (for example, slow movements of a robot arm and turntable).

The crucial part of the model is the automatic reconstruction. While it is easy to generate object models, the reconstruction of scenes depends on the scene itself and the taken image sequence. Thus, the main focus of our future work will be on the improvement of the calibration step. Also, hardware accelerated rendering techniques are necessary for future frame–rate applications.

References

[1] E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. In M. Laudy and J. A. Movshon, editors, *Computational Models of Visual Processing*, pages 3–20, Cambridge, MA, 1991. MIT Press.

[2] J.-X. Chai, X. Tong, S.-C. Chand, and H.-Y. Shum. Plenoptic sampling. *Proceedings SIGGRAPH 2000*, pages 307–318, July 2000.

[3] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry– and image–based approach. In *Proceedings*

SIGGRAPH '96, pages 11–20, New Orleans, August 1996. ACM Press.

[4] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999.

[5] A. S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers, San Francisco, CA, 1995.

[6] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Proceedings SIGGRAPH '96*, pages 43–54, New Orleans, August 1996. ACM Press.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[8] B. Heigl. *Plenoptic Scene Modeling from Uncalibrated Image Sequences*. PhD thesis, University of Erlangen–Nuremberg, 2003. to appear.

[9] B. Heigl, J. Denzler, and H. Niemann. On the application of lightfield reconstruction for statistical object recognition. In *European Signal Processing Conference*, pages 1101–1105, 1998.

[10] B. Heigl, J. Denzler, and H. Niemann. Combining computer graphics and computer vision for probabilistic visual robot navigation. In Jacques G. Verly, editor, *Enhanced and Synthetic Vision 2000*, volume 4023 of *Proceedings of SPIE*, pages 226–235, Orlando, FL, USA, April 2000. SPIE–The International Society for Optical Engineering, Bellingham, WA.

[11] B. Heigl, R. Koch, M. Pollefeys, J. Denzler, and L. van Gool. Plenoptic modeling and rendering from image sequences taken by a hand–held camera. In W. Förstner, J. M. Buhmann, A. Faber, and P. Faber, editors, *Mustererkennung 1999*, pages 94–101, Heidelberg, September 1999. Springer–Verlag.

[12] R. Koch, B. Heigl, M. Pollefeys, L. van Gool, and H. Niemann. A geometric approach to lightfield calibration. In F. Solina and A. Leonardis, editors, *Computer Analysis of Images and Patterns — CAIP '99*, number 1689 in Lecture Notes in Computer Science, pages 596–603, Ljubliana, Slovenia, 1999. Springer–Verlag.

[13] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings SIGGRAPH '96*, pages 31–42, New Orleans, August 1996. ACM Press.

[14] L. McMillan. *An Image–Based Approach to Three–Dimensional Computer Graphics*. PhD thesis, Department of Computer Science, University of North Carolina at Chapel Hill, Chapell Hill, North Carolina, 1997.

[15] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):206–218, March 1997.

[16] S. M. Seitz and C. R. Dyer. View morphing. In *Proceedings SIGGRAPH '96*, pages 21–30, New Orleans, August 1996. ACM Press.

[17] J. Shi and C. Tomasi. Good features to track. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, Seattle, WA, June 1994. IEEE Computer Society Press.

[18] P. Sturm and B. Triggs. A factorization based algorithm for multi–image projective structure from motion. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 709–720. Springer–Verlag, 1996.

[19] C. Vogelgsang, B. Heigl, G. Greiner, and H. Niemann. Automatic image–based scene model acquisition and visualization. In B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, editors, *Workshop Vision, Modeling and Visualization*, pages 189–198, Saarbrücken, Germany, November 2000. Akademische Verlagsgesellschaft Aka GmbH, Berlin.

[20] D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin, and W. Stuetzle. Surface light fields for 3D photography. In *Proceedings SIGGRAPH 2000*, New Orleans, July 2000. ACM Press.

Bild

Joachim Denzler graduated 1992 with the degree 'Diplom-Informatiker' from the University Erlangen-Nürnberg, Germany. He received his doctoral degree in computer science in 1997. Currently he is head of the computer vision group of the Institute for Pattern Recognition of the University Erlangen-Nürnberg. His research activities concentrate on probabilistic modeling of sensor data and action sequences in the field of computer vision.

Bild

Benno Heigl received the degree 'Diplom-Informatiker' in computer science from the University Erlangen-Nürnberg, Germany, in 1996 and afterwards he joined the Institute for Pattern Recognition until March 2000. Currently he is working at Siemens Medical Solutions, Forchheim, Germany, dealing with 3-D reconstruction from angiographic x-ray images. His research interest include camera calibration, structure from motion, and plenoptic scene modeling.

Bild

Matthias Zobel graduated in 1998 with the degree 'Diplom-Informatiker' at the University Erlangen-Nürnberg, Germany. Since 1998 he is with the Institute for Pattern Recognition at the same university, where he is a research member at the SFB 603 "Model-Based Analysis and Visualization of Complex Scenes and Sensor Data". His research is especially about multiocular object tracking with optimal adaption of the camera parameters, and about sensor data fusion.

Bild

Heinrich Niemann obtained the degree of Dipl.-Ing. in Electrical Engineering and Dr.-Ing. from Technical University Hannover, Germany. Since 1975 he has been Professor of Computer Science at the University of Erlangen-Nürnberg. Since 1998 he is speaker of a 'special research area' (SFB) entitled 'Model–Based Analysis and Visualization of Complex Scenes and Sensor Data' which is funded by the German Research Foundation (DFG). His fields of research are speech and image understanding and the application of artificial intelligence techniques in these fields.