PAPER *Special Issue on Machine Vision Applications*

Calibration of Real Scenes for the Reconstruction of Dynamic Light Fields*

Ingo SCHOLZ[†], **Joachim DENZLER**[†], ***and*** **Heinrich NIEMANN**[†], ***Nonmembers***

SUMMARY

The classic light field and lumigraph are two well–known approaches to image–based rendering, and subsequently many new rendering techniques and representations have been proposed based on them. Nevertheless the main limitation remains that in almost all of them only static scenes are considered. In this contribution we describe a method for calibrating a scene which includes moving or deforming objects from multiple image sequences taken with a hand–held camera. For each image sequence the scene is assumed to be static, which allows the reconstruction of a conventional static light field. The dynamic light field is thus composed of multiple static light fields, each of which describes the state of the scene at a certain point in time. This allows not only the modeling of rigid moving objects, but any kind of motion including deformations.

In order to facilitate the automatic calibration, some assumptions are made for the scene and input data, such as that the image sequences for each respective time step share one common camera pose and that only the minor part of the scene is actually in motion.

key words: *Dynamic light field, camera calibration, structure from motion, image–based rendering*

1. Introduction

In recent years the field of image–based rendering has become a very popular research topic. The light field [7] and the lumigraph [3] are two similar and often used approaches for modeling objects or scenes from a set of input images and without prior knowledge of scene geometry. One of their advantages over conventional model–based rendering techniques is that they allow photo–realistic rendering of real scenes or objects, while computation time is independent of the complexity of scene geometry.

While it is already possible to generate light fields from real but static scenes and render high–quality images from them [6], these light fields are not applicable to dynamic scenes, i.e. scenes that vary over time. Nevertheless a lot of applications can be thought of where dynamic light fields would be useful. For instance in endoscopic, minimal–invasive surgery [18] an automatically generated light field would allow the physician to view the organ he is operating from any view point without having to move the endoscope, thus reducing the strain on the patient. The problem in such an application is that the position or shape of organs may change during an operation and that an organ is in permanent motion. The static light field would thus be insufficient to model such a scene.

We currently focus our research on solutions for applications like the above, which can be generally described as real scenes containing moving and deforming objects. At present we also require the scene to have a static background, while the dynamic part of the scene is smaller than the rest. The extension of static light fields to dynamic scenes and objects gives rise to several problems:

- Calibration of scenes which include moving objects has to cope with unreliable point correspondences, requiring the identification of different time frames and the distinction of static and dynamic parts of the scene.
- The amount of data to be stored is already large for static light fields, but dynamic light fields further increase the dimension of the parameter space.
- Extended rendering techniques are required for rendering images at arbitrary points in space *and* time.

In this contribution we will concentrate mostly on the first of these issues: the reconstruction of a dynamic light field by calibrating a dynamic scene. Instead of using a calibration pattern like in [3], or placing the camera at known positions [7], we pursue the approach described in [6], which is to automatically calibrate the camera parameters of image sequences taken with a hand–held camera using structure from motion algorithms. The required point correspondences are established by automatically extracting and tracking point features in the scene. The most important issue which will be addressed in the following is the adaptation of these well-established methods to scene content which is changing over time.

The main problem in automatically calibrating dynamic scenes is that it is not possible to determine whether the movement of a point feature from image to image is due to the movement of the camera or of an object in the scene. For being able to use the latter points for calibration, the deformation of the scene itself

Manuscript received March 28, 2003.

Manuscript revised July 25, 2003.

Final manuscript received September 10, 2003.

[†]The authors are with the Faculty of the University Erlangen-Nürnberg, Chair for Pattern Recognition, Martensstr. 3, 91058 Erlangen, Germany

[*]This paper was presented at the IAPR Workshop on Machine Vision Applications 2002, Nara, Japan

would have to be known very precisely. Unfortunately, this knowledge cannot be gained if the camera movement is unknown. In order to break out of this vicious circle we assume that for each time step an individual image sequence is available. Each of these sequences now shows only a static scene, for which the calibration can be solved. The dynamic light field is then composed of multiple static light fields, one for each time step. Since the light field model requires a very precise reconstruction especially of the camera parameters, the registration of these static light fields has to be very accurate. The desired accuracy is reached by a refinement step following the first registration. A method for rendering images from the resulting dynamic light field from arbitrary viewpoints in space and time is introduced as well.

The modeling of dynamic scenes and objects is currently a very active topic of research. Solutions have been proposed for handling multiple rigid moving objects in a scene [2], [4], [9] or modeling non–rigid objects [1], [15]. While good results are already reached here, these approaches still need to constrain the underlying projection models or the type of object movement. In our approach on the other hand we can rely on the relative robustness and quality of established methods for calibrating static scenes, while the modeling of dynamics is done through the combination of the results.

The registration step will be described in detail in the next section, followed by a section treating the process of image rendering. Experimental results will be given in Section 4, and Section 5 offers some concluding remarks and an outlook to future work.

2. Calibrating Dynamic Light Fields

Instead of putting together a dynamic scene from one static light field for each rigid but moving object, as it was described in [8], we subdivide the dynamic scene into k time steps and model each with a complete static light field of the scene. We are thus able to not only model rigid but also deformable objects in the scene.

The input images we will use in the following to reconstruct a dynamic light field need to fulfill two main requirements. First, one image sequence must be available for each time step so that the k static light fields can be reconstructed from them. Second, for two consecutive image sequences the last camera of the first sequence must have approximately the same pose as the first camera of the second sequence, which means that the two sequences have one camera position in common. In practice, the camera is moved over the scene while the objects in it stay immobile, and is kept steady while the objects are moved to the next position. This second part, where the objects are moved, is cut away, leaving the desired input image sequences. Thus, the images seen from the common camera position of two sequences show a different state of the moving object in the scene. This different image content poses the main difficulty for registration.

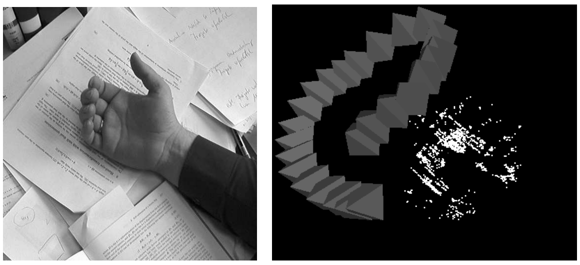


Fig. 1 Left: First image of the *Hand* image sequence. Right: Calibration results for this image sequence. Each pyramid represents a camera, the dots are the 3–D points on the scene surface.

The dynamic light field is reconstructed from this input data by first calibrating the individual image sequences and then registering the resulting threads of camera positions with each other. Finally a refinement step can be applied which calibrates all cameras together. The assumption which must be made for this last step to work is that dynamic objects only influence the lesser part of the visible scene. This means that the background of the scene covers the major part of each image so that more point correspondences are found on static parts of the scene than on moving ones. The three calibration steps will be described in the following.

2.1 Static Light Field Calibration

Each image sequence is calibrated independently following the approach of [5] which involves a three–step process. The first step is the establishment of point correspondences between the images through feature tracking. The feature selection and tracking algorithm used is a differential method proposed by Tomasi and Kanade [14], which detects point features based on the assumption that those features can be tracked best whose most significant gradients are perpendicular to each other. While tracking a feature window, the corresponding feature in the next image is selected which minimizes an error measure between the two images. The tracked window is validated using the extension by Shi [12] which calculates an affine distortion between the two windows.

In the second step an initial subsequence is calibrated using a hierarchy of different factorization methods. A weak–perspective factorization [10] is applied first which is very robust but yields imprecise results. These can be used as an initialization for a following projective factorization [13], which is more sensitive to erroneous point correspondences but leads to a more correct result due to the underlying, more realistic projection model.

In the last step the remaining cameras are added by using the reconstructed 3–D points as a calibration pattern. By triangulating features in previously calibrated images they can be used as 3–D correspondences to the 2–D features in uncalibrated images and calibration methods as in [16], [17] and can be applied thereon. This way, it is possible to calibrate even long image sequences of up to a thousand images. The portion of images in the initial subsequence used for factorization depends on the average number of images the point features are seen in. For example it is possible to calibrate a sequence of 500 images with only 20 images in the initial subsequence. The method is described in more detail in [5].

Apart from the projection matrix of each camera the calibration also yields a set of 3–D points which correspond to the 2–D feature points used for calibration. The cameras and 3–D point sets for the calibration of an example image sequence are shown in Figure 1. In this representation, each camera is symbolized as a pyramid where the position of its projection center is at the top of the pyramid and its image plane corresponds to the base of the pyramid.

The coordinate systems of the reconstructed cameras of two image sequences now differ from each other by a rotation, a translation and an unknown scale factor, and need to be registered with each other in the following steps.

2.2 Registration

The rotation and translation can be determined by the fact that two cameras of a pair of consecutive image sequences have approximately the same pose. The transformation is done by first mapping one of the two cameras into the origin of its coordinate system and then to the pose of the other camera. If the 3×4 projection matrix of the second camera is given as $\mathbf{P}_2$, and the rotation matrix and translation vector of the two cameras as $\mathbf{R}_1$, $\mathbf{R}_2 \in \mathbb{R}^{3\times3}$ and $\mathbf{t}_1$, $\mathbf{t}_2 \in \mathbb{R}^3$ respectively, the transformation is done as follows:

$$\mathbf{P}'_2 = \mathbf{P}_2 \begin{pmatrix} \mathbf{R}_2^T & -\mathbf{R}_2^T\mathbf{t}_2 \\ \mathbf{0}_3^T & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{R}_1^T & -\mathbf{R}_1^T\mathbf{t}_1 \\ \mathbf{0}_3^T & 1 \end{pmatrix}. \quad (1)$$

The result $\mathbf{P}'_2$ denotes the new projection matrix in the coordinate system of the first sequence. The same transformation is then applied to all remaining $M-1$ cameras of the second sequence, denoted in the following by $\mathbf{P}'_{m2}, m \in \{2, \ldots M\}$. The reverse of this transformation,

$$\mathbf{p}'_{n2} = \begin{pmatrix} \mathbf{R}_1^T & -\mathbf{R}_1^T\mathbf{t}_1 \\ \mathbf{0}_3^T & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{R}_2^T & -\mathbf{R}_2^T\mathbf{t}_2 \\ \mathbf{0}_3^T & 1 \end{pmatrix} \mathbf{p}_{n2}\,, \quad (2)$$

is applied to each of the N 3–D points $\mathbf{p}_{n2}, n \in \{1, \ldots N\}$, of the same sequence to convert them in the same way. All vectors are given in homogenous coordinates, so, in order to get the Euclidian coordinates, they have to be normalized by division by the last entry.

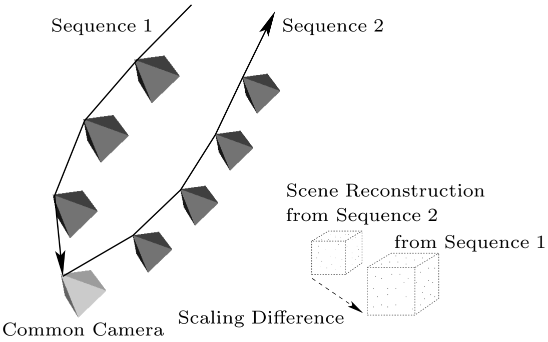


Fig. 2 Registration of two image sequences. The incorrect scaling results in different distances of the reconstructed point clouds of the scene from the cameras.

Figure 2 depicts such a registration of two image sequences using a common camera. It also shows the effect of the missing scaling step: on the one hand it results in different distances of the two 3–D point clouds, one for each image sequence, from the camera positions. On the other hand, the distances between the cameras are scaled by the same factor, which is indicated by smaller cameras for the second sequence in addition to the smaller spacing between them.

The scale factor is obtained by considering the centers of mass of the 3–D points in each image sequence. As the sequences were taken of the same scene, the centers of mass are assumed to be 3–D points which are roughly at the same position in the scene. If the camera is moved on approximately the same path for two consecutive image sequences this assumption holds, since the features selected by the tracking algorithm will be similar. The scale factor s is computed as the ratio of the distances of the centers of mass from the two equal cameras of two consecutive sequences.

Once this ratio is known all cameras and 3–D points of the respective second sequence can now be scaled to be registered correctly with the first sequence. Again, the projection matrices of all cameras are first transformed such that the common (first) camera is moved to the origin of the coordinate system, scaled by a matrix

$$\mathbf{S} = \begin{pmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

and then moved back by the reverse first transformation. This way, the first camera stays at the same position, which is equal to that of its counterpart in the first sequence. Thus, this transformation can be written as follows:

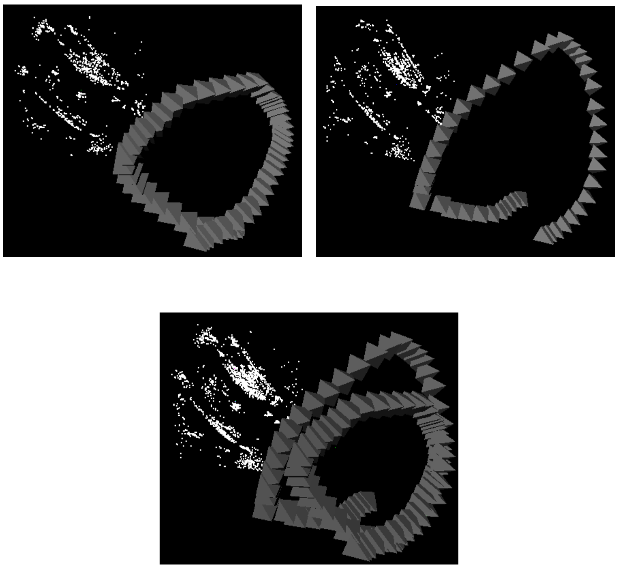


Fig. 3 Example for the registration of image sequences. Top Row: Calibration of two image sequence showing the camera positions as pyramids and 3–D point clouds in the background. Bottom: Registration of the two image sequences.

$$\mathbf{P}''_{m2} = \mathbf{P}'_{m2} \begin{pmatrix} \mathbf{R}_2^T & -\mathbf{R}_2^T \mathbf{t}_2 \\ \mathbf{0}_3^T & 1 \end{pmatrix}^{-1} \mathbf{S} \begin{pmatrix} \mathbf{R}_2^T & -\mathbf{R}_2^T \mathbf{t}_2 \\ \mathbf{0}_3^T & 1 \end{pmatrix} . \tag{4}$$

It is applied again to each of the m cameras in the second sequence. $\mathbf{R}_2$ and $\mathbf{t}_2$ without the index m denote the rotation matrix and translation vector of the first (common) camera.

The scaling of the 3–D points is done analogously to equation (2).

Up to now we did not consider a possible variation of the intrinsic parameters of the cameras, like e. g. the focal length. These parameters are also assumed to be constant throughout the scene recording. Small changes due to the automatic focus adaption of the camera proved to be tolerable. The equality of the intrinsic parameters for the independent calibrations of two sequences is given if the same initial values are set during factorization.

A final result of a registration of two image sequences is shown in Figure 3. These image sequences, again symbolized by one pyramid for each camera, are again of the *Hand* scene which was introduced in Figure 1. The 3–D points on the scene surface can be seen in the background of the images as white dots. The two images in the top row show the calibration of two individual image sequences, and in the bottom image, they were registered using the method described above. As it can be seen in the similarity of the point clouds in both upper images, the requirement for the calculation of the scale factor that both sequences see approximately the same patch of the scene is fulfilled here. As a result, the two point clouds are well registered in the lower image.

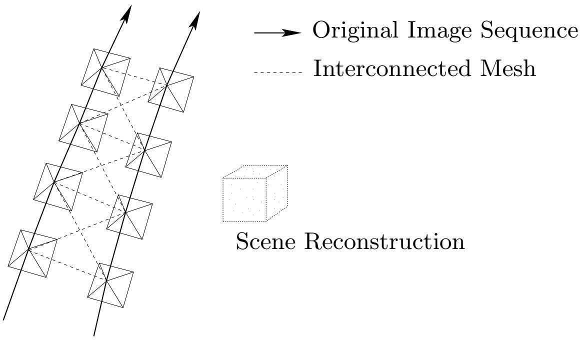


Fig. 4 Creating a mesh between the original image sequences. Two different sequences are sketched along the solid line, while the detected neighbourhood relations are drawn as dashed lines.

For a more concise appearance of the data the camera sequences in Figure 1 and Figure 3 were thinned out to show only a quarter of the cameras in the first case and half of the cameras in the second.

2.3 Refinement

After transformation of the cameras of all light fields to the same coordinate system, a further refinement of the calibration is performed. The camera positions form a 3–D mesh in which neighbours with similar views on the scene can easily be identified. In order to make sure that the corresponding images show similar parts of the scene the viewing direction of two neighbouring cameras must be similar, too. Thresholds for the maximum distance and viewing direction difference for two cameras are calculated as multiples of the average values of all pairs of subsequent cameras.

In Figure 4 this process is shown schematically. Parts of two image sequences are shown on the left side along solid lines with four cameras each. The neighbourhood connections that could be established automatically are denoted by dashed lines, provided that these cameras are close enough to one another.

Using these neighbourhoods a second tracking step is invoked. This time, no new features are added but only those are tracked further that were used for the first calibration. The calibration process removes outliers by discarding features with a too high back–projection error and features that could only be tracked over 2 or 3 frames, leaving only the more robust ones.

Tracking is performed in a two–step loop for each image sequence:

1. The existing features in the current sequence are tracked to the other image sequences following the neighbourhood links established before. The tracking algorithm is implemented in such a way that features can be tracked from one image to any other image available. No reordering of the image sequences is required.

2. These additional features are now propagated through the other image sequences. Depending on the effort to be spent this can be just the preceding and the following image sequence of the current one, or all other image sequences. The complexity in the latter case of course increases quadratically with the number of time steps.

Through this obviously time–consuming process, the formerly mostly unrelated sequences—except for the one frame which is common in each pair of consecutive sequences—are now linked together through these new feature correspondences.

In a second calibration step the camera parameters for the whole set of images of all sequences can now be calculated together. Like for each individual image sequence before, an initial subset of images is now searched which has the highest number of feature correspondences in common. Unlike before, this subset does not need to be a subsequence of consecutive frames anymore, but constitutes a subgraph of the neighbourhood mesh constructed for the feature tracking step (see Figure 4). Calibration is then performed as before, using a factorization for the initial subset and adding frames one by one through conventional calibration (see Section 2.1). Which frame to add next is again determined by the neighbourhood mesh.

Afterwards, all camera parameters are available in the same coordinate system, and no second refinement step is necessary. The error which occurs if the corresponding cameras of each pair of sequences were not exactly the same after all—which is usually the case since the hand holding the camera trembles while the object is moved—is thus removed.

The disadvantage of the new feature correspondences is that any of them could be positioned on a moving, i. e. dynamic, part of the scene. These *dynamic features* can be considered equivalent to erroneously tracked points, and can severely perturb the calibration process. Nevertheless, by postulating that only a minor part of the scene is actually in motion the calibration algorithm proved to be robust enough to handle these outliers.

The reason is that after the factorization of a subset of only a few cameras (see Section 2.1) the calibration is extended to the rest of the sequences by classical calibration techniques [16]. In this step the 3–D points acquired through factorization are used again as calibration pattern, which is extended after calibration of each new image by triangulating the features found in it. Now 3–D points with a high back–projection error are discarded, which is often the case for dynamic features from two different image sequences.

On the other hand, features on a dynamic part of the scene are unproblematic as long as they are not tracked to another image sequence at a different time step.

3. Rendering

Since we are using a hand–held camera for capturing the images for our dynamic light fields, the camera positions may be distributed almost arbitrarily in space. Therefore the most obvious parameterization is that of a free form light field [6]. The input images and their camera positions are stored as is in the light field structure and require no further processing. New views are generated by selecting the radiance values for interpolation from the three cameras closest to the currently considered viewing ray.

Other parameterizations, like the two planes first proposed in the original lumigraph and light field papers [3], [7], require the camera positions and image planes to be positioned on two regular grids. If this is not the case for the original images, a warping step is applied to them first which can significantly decrease image quality.

For rendering images from a dynamic light field we extended our already existing hardware–accelerated free form renderer [11] to handle an arbitrary number of static light fields. The difference to rendering images from static light fields is that a timestamp is now required as an additional parameter. Rendering images at known time steps, i. e. those where the image sequences were taken, can thus be done without additional effort.

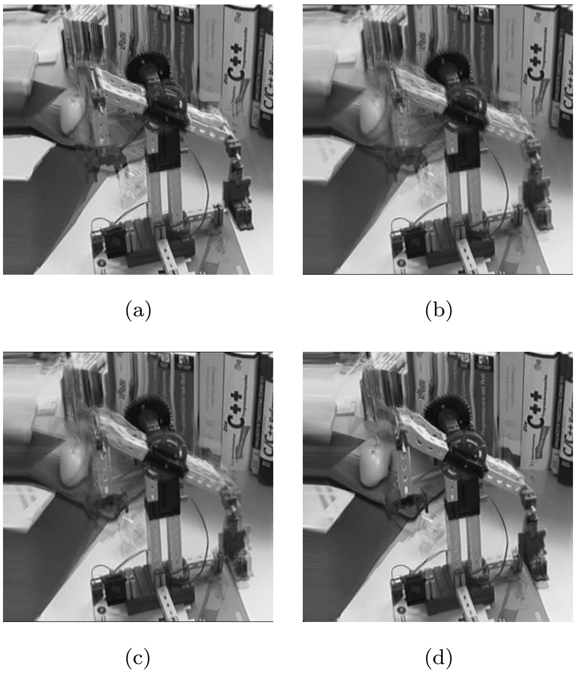
Generating views of the scene at arbitrary positions in time on the other hand is a much more difficult problem, and many different solutions can be thought of. One approach is to first render the views for the earlier and later integer time steps and then generate images at any intermediate time by interpolation.

Since the emphasis of our current work is not on the rendering of light fields but on their generation, we only implemented the basic technique of creating new views by cross–fading the two available images, weighted by their distance to the desired timestamp. The result can be seen in Figure 5, where images (a) and (d) are of two subsequent integer time steps, and images (b) and (c) the two steps in between.

Using additional information about the scene, which is already available through the calibration process, more sophisticated methods can be applied as well. The back–projections of the known 3–D points into the rendered image can be used as control points for the application of different kinds of image warping techniques which are widely used in computer graphics [19].

4. Experiments

The experiments described in the following section were conducted to analyze the quality of registration of the image sequences. For this purpose three dynamic light fields were created as described in Section 2. Each of



(a) (b)

(c) (d)

Fig. 5 Light field of a toy rotor. Images (a) and (d) are of integer time steps, (b) and (c) show intermediate steps generated by cross–fading.

them shows a different dynamic object in front of a static background. Rendered example images of each scene are shown in Figure 6. Column 2 of Table 1 states the number of available image sequences (time steps) for each scene. The number of images per image sequence varies between 64 and 108, so that the total number of frames for the light fields are 436, 482 and 576 for *Hand*, *Head* and *Rotor* respectively.

One error measure which is often used when calibrating image sequences is the back-projection error of point features. The calibration process yields a number of 3-D points whose corresponding 2-D features in some images are known. The recovered projection matrices can be used to project the 3-D points into these images and the result is compared with the 2-D features. The resulting distance depends on the quality of calibration, but also on the quality of feature detection and tracking.

An assessment of the quality of registration using the back-projection error is shown in Figure 7 for the *Rotor* sequence. In order to ensure the comparability between registration by concatenation and the following refinement, the same set of point correspondences was used for both. These consist exclusively of features which were observed in more than one of the image sequences. Unlike described above the corresponding 3-D points were not taken from the calibration process, but they were triangulated from the 2-D points using a non-linear optimization method.

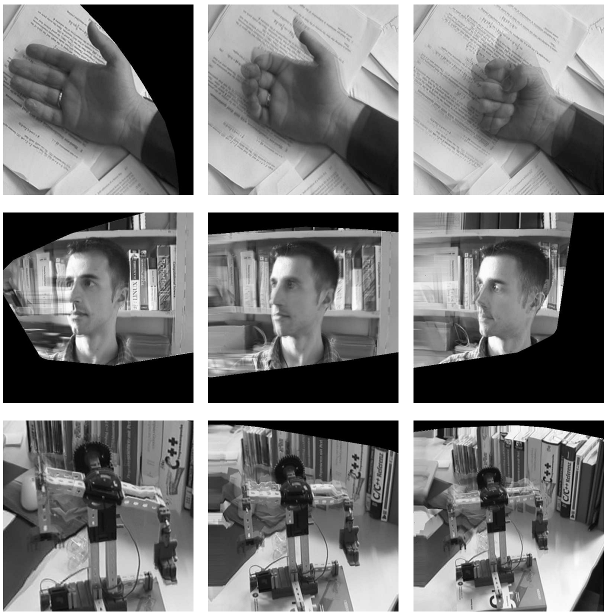


Fig. 6 Example images from the three scenes *Hand*, *Head* and *Rotor* used for the experiments. The images in the first two rows were rendered using a constant camera pose, while for the third row the camera was moved and zoomed simultaneously.

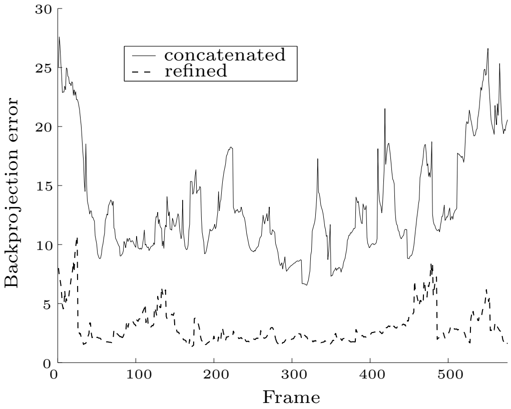


Fig. 7 Backprojection error of point correspondences with features in images of at least two sequences. The error in each frame of the *Rotor* sequence is given in pixels, before (solid line) and after (dashed line) the refinement step.

The comparison in Figure 7 shows clearly that the backprojection error is considerably smaller in each image of all sequences after the refinement than before. The average pixel error in the *Rotor* example was reduced from 13.3 to 3.0. For *Hand* and *Head* sequences the average error was reduced from 4.0 to 1.0 and from 8.8 to 1.3 pixels respectively. It should be stated here that the back-projection errors are usually smaller during calibration since not all of the point correspondences are used there. If the error of a 2-D point is too high it is discarded.

Another measure for the quality of image sequence registration is the shift of the background for two rendered images of different time steps, but as seen with

		# Image diffs		Mean diff	
Scene	# Seq.	concat	refine	concat	refine
Hand	5	10	10	17.3	12.0
Head	6	18	15	30.8	17.6
Rotor	8	30	28	40.3	12.7

Table 1 Comparison of background shifts in the example scenes using mean pixel difference. Columns 3 and 4 denote the number of image comparisons which were performed, columns 5 and 6 show the average pixel difference of all image pairs in each light field.

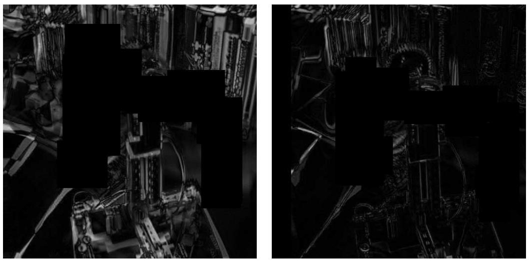


Fig. 8 Difference images for time steps 4 and 7 of the *Rotor* sequence from similar camera positions before and after refinement.

exactly the same virtual camera pose. Putting this shift into numbers is difficult, and we chose the average absolute pixel difference as a measure. While this lacks some quantitative expressiveness, it can still give a good qualitative impression.

Since only the background shift was to be considered, the dynamic objects in the foreground were removed by hand–coloring them in black. These and any other colorless parts of the images—they appear if no reference images for interpolation are close enough to this area—were ignored in the difference. Columns 5 and 6 of Table 1 show the pixel differences for the simple image sequence concatenation of Section 2.2 compared to the value after the refinement described in Section 2.3. In all scenes the refinement step clearly improves the registration. This can be seen in the example in Figure 8 where the difference images of two time steps are plotted, before (left) and after the refinement (right).

In order to ensure the validity of this comparison the test images were always rendered with approximately the same visible object sizes. An exact match was not possible since the coordinate systems differ before and after refinement. The values in columns 3 and 4 of Table 1 denote the number of image comparisons performed. These may vary since for some viewpoints the renderer was not able to generate valid images for every time step. This may happen if the chosen viewpoint is too far from the camera positions of the input images. Column 3 often contains higher values than column 4 since this problem occurred more often for the normal registration. In those cases additional viewpoints were selected.

5. Conclusion and Future Work

In the preceding sections we described a method for reconstructing a dynamic light field from multiple image sequences, each referring to a different time step. The preconditions are that image sequences of consecutive time steps share a common frame concerning the camera pose and parameters, and that only the lesser part of the scene is in motion. By calibrating each image sequence independently and concatenating the results, a good registration of the camera sequences can be achieved. A subsequent refinement step can further improve the quality of registration.

By storing each time step as an individual light field the modeling of arbitrary movements of the scene is possible. Images at intermediate time steps can be rendered by cross–fading images from neighbouring known light fields.

Our future research will focus on the relaxation of the above requirements, so that the object can be in motion while being recorded. Therefore we currently examine factorization algorithms which are able to separate camera and scene motion using rank constraints. Examples for these have already been mentioned in the introduction ([1], [4]). The results could be used as initialization of an iterative method for alternately updating camera pose and scene reconstruction.

In addition to that the two issues of rendering images at any point in time and of efficiently storing dynamic light fields have to be addressed in the future.

Acknowledgement

This work was funded by the German Research Foundation (DFG) under grant SFB 603/TP C2. Only the authors are responsible for the content.

Thanks to Christian Vogelgsang for his support on the free form light field renderer which was also developed under the above grant.

References

[1] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *Proc. of IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 690–696, 2000.

[2] J. P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

[3] 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.

[4] M. Han and T. Kanade. Multiple motion scene reconstruction with uncalibrated cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):884–894, July 2003.

[5] B. Heigl. *Plenoptic Scene Modeling from Uncalibrated Image Sequences.* 2003. Dissertation, Faculty of Engineering Sciences, University Erlangen–Nürnberg, to appear.

[6] 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 *Mustererkennung 1999*, pages 94–101, Heidelberg, 1999. Springer–Verlag.

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

[8] W. Li, Q. Ke, X. Huang, and N. Zheng. Light field rendering of dynamic scene. *Machine Graphics and Vision*, 7(3), 1998.

[9] R. Manning and C. Dyer. Affine calibration from moving objects. In *Proceedings of the 8th IEEE International Conference on Computer Vision*, volume I, pages 494–500, 2001.

[10] 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.

[11] H. Schirmacher, C. Vogelgsang, H.-P. Seidel, and G. Greiner. Efficient free form light field rendering. In *Proceedings of Vision, Modeling, and Visualization 2001*, pages 249–256, Stuttgart, Germany, November 2001.

[12] 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.

[13] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure from motion. In *Proceedings ECCV*, pages 709–720. Springer, 1996.

[14] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, 1991.

[15] L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *Proceedings of IEEE Conference Computer Vision and Pattern Recognition*, 2001.

[16] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision.* Addison–Wesley, Massachusets, 1998.

[17] R. Y. Tsai. A versatile camera calibration technique for high–accuracy 3D machine vision metrology using off–the–shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, RA–3(4):323–344, August 1987.

[18] F. Vogt, D. Paulus, and H. Niemann. Highlight Substitution in Light Fields. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 637–640, Rochester, USA, September 2002. IEEE Computer Society Press.

[19] G. Wolberg. *Digital Image Warping.* IEEE Computer Society Press, 1990.

Ingo Scholz Ingo Scholz studied computer science at the University Erlangen–Nürnberg, Germany, between 1994 and 2000, with an emphasis on pattern recognition and image processing. He graduated with the degree 'Diplom-Informatiker'. Since 2001 he is working as a research staff member at the Institute for Pattern Recognition of the University Erlangen–Nürnberg. His main research focuses on the reconstruction of light field models, camera calibration techniques and structure from motion. He is a member of the German 'Gesellschaft für Informatik' (GI).

Joachim Denzler Joachim Denzler studied computer science, especially pattern recognition, speech recognition and theoretical electrical engineering at the University Erlangen–Nürnberg, Germany, from 1987 to 1992 and graduated with the degree 'Diplom-Informatiker'. He received his doctoral degree in computer science in 1997, and the 'Habilitation' in June 2003. From January 1993 to August 2003 Joachim has been member of the research staff of the Institute for Pattern Recognition of the University Erlangen–Nürnberg, and since 2002 head of the computer vision group. Currently he holds the position of a professor at the department of mathematics and computer science at the University of Passau, Germany. His research activities concentrate on probabilistic modeling of sensor data and action sequences in the field of computer vision. Joachim is member of the IEEE, IEEE Computer Society and GI.

Heinrich Niemann Heinrich Niemann obtained the degree of 'Diplom-Ingenieur' 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.