Vision, Modeling, and Visualization 2004, Aka / IOS Press, Berlin, Amsterdam, pages 109–116

Extending Light Fields using Object Tracking Techniques

Benjamin Deutsch *, Ingo Scholz *, Christoph Gräßl †, Heinrich Niemann

Lehrstuhl für Mustererkennung
Universität Erlangen-Nürnberg
Martensstr. 3, 91058 Erlangen, Germany
Email: {deutsch,scholz,graessl,niemann}@informatik.uni-erlangen.de

Abstract

We present two new approaches to extending existing light fields with additional image data. In this case a light field is initially constructed from an image sequence taken by a hand-held camera, and pose parameters of this camera obtained through structure-from-motion approaches. To extend such a light field, point correspondences are necessary from one image in the original sequence to the new images to estimate their relative poses. The two introduced approaches assist in finding the original image closest to the new image, and provide initial motion estimates. A SIFT feature based method is used to determine the closest image and an image-space motion homography. The second approach uses images rendered from the light field to estimate the camera pose of the image to be added using adaptive random search or a particle filter.

1 Introduction

The light field was first introduced in [9] and similarly in [3] as a means to render arbitrary views of a previously recorded real scene without requiring a geometric model of the scene. The various different approaches to light field rendering nevertheless range from purely image-based [9] to those applying at least local depth information for each input image [3, 2].

A light field requires, besides the input images and the already mentioned depth information, knowledge about the pose parameters of the recording camera for each image. In many applications these parameters can be acquired by a mechanical arm or gantry. A more flexible and inexpensive approach is the use of a hand-held camera and the recording of continuous image sequences [7]. The pose parameters are then estimated using structure-from-motion algorithms such as factorization methods [15, 12] and non-linear optimization [6]. This process was applied for all light field computations throughout this contribution.

For this kind of camera parameter reconstruction point feature correspondences between the images of the sequences have to be calculated. However, many commonly used feature tracking algorithms [14, 18] assume that camera motion between two consecutive images is quite low and that the search range can thus be restricted considerably. The drawback of this assumption is that once the recording of an image sequence has been completed it is difficult to track features in any new images or image sequences which were taken later from a different view point and thus calculate the corresponding camera pose parameters. For the light field this means that once it has been reconstructed from one image sequence, it is difficult to extend it e. g. to cover a broader viewing range or to improve its quality by adding more images.

Therefore, in order to allow this extension of a light field with new images it is necessary to at least find the most similar already known image to the new one. Since the disparity between those two images may be too large for robust feature tracking, it is desirable to additionally supply a good estimate for the new feature positions.

In this contribution we present two methods which are able to determine both the closest image and a feature position estimate. The first one is based on local features, namely David Lowe's SIFT features [10], which has been proven to be a very efficient technique in a quantitative comparison of different local descriptors [11]. The most

*This work was partially funded by the German Research Foundation (DFG) under grant SFB 603/TP B2 and C2. Only the authors are responsible for the content.

†This work was partially funded by the European Commission 5th IST Programme - Project VAMPIRE. Only the authors are responsible for the content.

similar image is found by using a majority voting technique, i. e. counting the number of best matching SIFT features. Once the most similar image is found, the SIFT features are used to estimate the 2-D homography by a least squares approach for translating the pixels from the matched image to the new image. Using this homography, the point tracker is able to continue tracking on the new image.

The second method uses the rendered images from the light field constructed so far as feedback for a parameter search. The camera parameters (position and orientation) are optimized such that the image rendered from them is most similar to the new image to be inserted. We use two optimization methods: adaptive random search [16], a robust global optimizer, and a particle filter [8] based approach. These methods can perform a global search, or can use the image determined by the SIFT feature method above as a starting point. The resulting camera parameters are then used as a starting point for inserting new images into the light field.

The remainder of this contribution is structured as follows. Section 2 gives a short introduction to the methods used for camera motion estimation and thus light field reconstruction in general. The requirements for extending the light field with new images are described as well. The image matching algorithm and homography computation using SIFT features is discussed in Sect. 3, and Sect. 4 concentrates on the probabilistic camera pose estimation with particle filters. An experimental evaluation of the algorithms follows in Sect. 5, the article is concluded by a summary and an outlook to future work.

2 Light Field Reconstruction

A light field in its basic form constitutes a collection of light rays emitted from the scene surface. Each input image contributes a set of light rays to this collection, and a new image is rendered by interpolating between the closest known light rays for each pixel. In actual implementations [13, 2] this interpolation is often done on triangle patches instead of individual pixels.

In order to reconstruct a light field from an image sequence it is thus necessary to compute the origin and direction of the light rays represented by the pixels in each image. This is done by estimating the pose and projection properties of the camera for each image and an associated depth map. The procedure is explained in the following, along with the requirements for extending an existing light field.

2.1 Camera Parameter Estimation

As mentioned before, the basis of many structure-from-motion algorithms is the availability of point correspondences for the different input images. In our case, these are computed using an extension of the Tomasi-Kanade algorithm [14] by Zinßer et al. [18]. Since these tracking algorithms perform well on continuous image sequences but poorly in case of high image disparity, they form the main problem for extending a light field with new images as it will be explained in Sect. 2.2.

The camera parameter estimation is done in two steps. In the first, the computation is done for only a sub-sequence $\boldsymbol{f}_s$ of all images $\boldsymbol{f}_i$, i. e. $s = i_a, \ldots, i_b$. A factorization method [15] is able to estimate the relative camera pose parameters $\widehat{\boldsymbol{R}}_s$, $\widehat{\boldsymbol{t}}_s$ for a set of images from point features which are visible in all images. For each feature $\boldsymbol{q}_{s,j}$ in image $\boldsymbol{f}_s$ a corresponding 3-D point $\widehat{\boldsymbol{p}}_j$ is returned as well. In our case the method by Poelman and Kanade [12] is used which assumes a paraperspective projection model.

For a Euclidean reconstruction the projection of a 3-D point into an image $\boldsymbol{f}_i$ is given by

$$\boldsymbol{q}_{i,j} = \boldsymbol{P}_i \boldsymbol{p}_j = \boldsymbol{K}_i (\boldsymbol{R}_i^T | - \boldsymbol{R}_i^T \boldsymbol{t}_i) \boldsymbol{p}_j \qquad (1)$$

where $\boldsymbol{K}_i$ contains the intrinsic parameters for camera i, focal length, pixel size ratio and center of projection, and $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$ the pose parameters [4]. Since the factorization does not yield $\boldsymbol{K}_i$, the intrinsic parameters are set to standard values which are only close to reality and equal for all camera positions. The estimates $\widehat{\boldsymbol{P}}_s = \boldsymbol{K}_s (\widehat{\boldsymbol{R}}_s^T | - \widehat{\boldsymbol{R}}_s^T \widehat{\boldsymbol{t}}_s)$ and $\widehat{\boldsymbol{p}}_j$ from the factorization are now refined by minimizing the back-projection error

$$\epsilon_s = \sum_j (\boldsymbol{q}_{s,j} - \widehat{\boldsymbol{P}}_s \widehat{\boldsymbol{p}}_j)^2 \qquad (2)$$

for each image $\boldsymbol{f}_s$ by optimizing in turn the camera pose parameters and the 3-D points. This method of computing a Euclidean reconstruction by non-linear optimization using the Levenberg-Marquardt algorithm was similarly proposed by Hartley [4]. In our case the reconstruction will be slightly skewed perspectively due to the inaccurate $\boldsymbol{K}_s$.

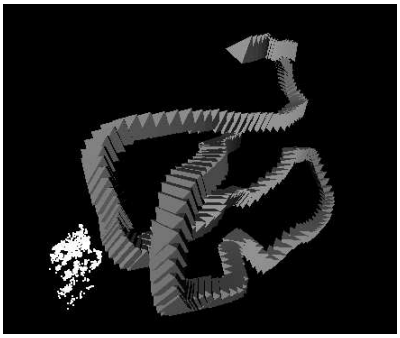

Figure 1: Camera pose and 3-D point reconstruction of the *santa* sequence with 207 images. Cameras are depicted as pyramids with their bases facing in viewing direction.

For the second processing step this method of estimating camera parameters by non-linear optimization was extended in [6] to cover the rest of the image sequence, too. For each remaining image $\boldsymbol{f}_r$, $r = 1, \ldots, i_a - 1, i_b + 1, \ldots N$ the estimates in equation 2 are initialized with the closest known projection matrix $\widehat{\boldsymbol{P}}_{r\pm1}$ and the 3-D points visible in image $\boldsymbol{f}_r$. New 3-D points are added by triangulation as soon as they are visible in enough known images. The result of the reconstruction of the *santa* sequence introduced in Sect. 5 is shown in Fig. 1.

In addition to the now known camera parameters, a light field requires some depth knowledge for each image, especially for sparse input data in case of a hand-held camera. A simple and fast way to compute these depth maps is to use the depth values of the reconstructed 3-D points and interpolate the remaining pixels as a distance weighted sum of the three closest known 3-D points.

2.2 Extending the Light Field

Extending an existing light field with the above method, i. e. adding new images or whole sequences, poses one major difficulty: where to start feature tracking. It is difficult to assure, and shall not be required, that the new image is similar to the last in the original image sequence. However, we will assume that the new image is taken within, or close to, the convex hull of the camera positions of the original sequence. Therefore, the task is to find the closest known image so that feature tracking may generate enough point correspondences for estimating the camera parameters of the new image.

This information of the most similar image will be supplied by the methods described in the following two sections. However they also supply additional information which is helpful for tracking as well as calibration. The SIFT feature matching yields a 2-D homography which describes the transformation from the closest to the new image. Adaptive random search and particle filter of Sect. 4 return a camera pose estimation which can be used to project known 3-D points into the new image and thus yield an estimate for feature tracking.

3 Matching with SIFT Features

Determining the most similar image to a new one will be done in the following by a full comparison with all known images already in the light field. For the matching of images, we propose to use local features, namely the SIFT features introduced by [10], which have been shown to be very efficient in a quantitative comparison [11].

3.1 Acquisition of SIFT Features

The SIFT feature points are detected by applying a scale selection mechanism based on differences of Gaussian smoothed images. The scale-space is built by convolving with a Gauss filter and downsampling after each octave, so that a pyramid-like data structure is obtained. The difference of the Gauss filtered images is computed by the difference of neighboring scales. After that, feature points are detected by searching for maxima with respect to the eight bordering pixels. In a second step, all the points which represent a maximum in scale-space are selected, by comparing the closest pixel at the next higher and next lower scale. In a third step, pixels which lie on edges are also ruled out, because such points are poorly determined.

In the next step, a significant SIFT feature vector $\boldsymbol{c}$ is calculated for every SIFT feature point. Therefore, the orientation of a region of size 16x16 around the SIFT feature point has to be adjusted to achieve invariance from rotation. In the scaled image, the magnitude and orientation is calculated as presented in [10] and used to create an orientation histogram. 36 bins were used for this and each pixel is weighted by its magnitude and by a Gaussian kernel. The region which is used for calculating vector $\boldsymbol{c}$ is rotated to the maximum of the orientation his-

togram. The SIFT feature vector itself is formed by using the orientation histograms. Therefore, the oriented region is divided into 4x4 subregions, and for every subregion an orientation histogram consisting of eight bins is calculated. Thus the feature vector has 128 elements altogether. For further details we refer to [10].

3.2 Image Matching

In our work, we want to estimate which of the images of the first sequence is most similar to the first image $\bar{\boldsymbol{f}}$ (test image) of the second sequence. For our purpose, a technique based on majority voting, which is described below, is well suited.

For every image $\boldsymbol{f}_i, i = 1, 2, \ldots, N$ in the first sequence, a set of SIFT features $\boldsymbol{C}_i = \{\boldsymbol{c}_i^1, \boldsymbol{c}_i^2, \ldots, \boldsymbol{c}_i^{N_i}\}$ is calculated. The value N_i depends on the number of SIFT features which were detected by the SIFT feature point detector in image $\boldsymbol{f}_i$. Similarly, we compute $\bar{\boldsymbol{C}} = \{\bar{\boldsymbol{c}}^1, \bar{\boldsymbol{c}}^2, \ldots, \bar{\boldsymbol{c}}^{\bar{N}}\}$ for the test image $\bar{\boldsymbol{f}}$. For counting the votes we use an accumulator

$$a_n = \sum_{k=1}^{\bar{N}} \delta(n - \operatorname*{argmin}_i \min_j d(\boldsymbol{c}_i^j, \bar{\boldsymbol{c}}^k)) \qquad (3)$$

where δ is the Kronecker delta function, a_n is the accumulator entry for image $\boldsymbol{f}_n$ and $d(\cdot, \cdot)$ is the Euclidian distance of two SIFT features. The index b of the best matching image can be retrieved from the accumulator by

$$b = \operatorname*{argmax}_n a_n. \qquad (4)$$

Thus $\boldsymbol{f}_b$ is the image with the most matching SIFT features with the SIFT features of $\bar{\boldsymbol{f}}$.

3.3 Homography Estimation

For the 3-D reconstruction, we use the feature point tracker of [14] to solve the correspondence problem. Those tracking feature points differ from the SIFT feature points as their matching can be done in real-time [18] and scale space stability is not needed. But if one would try to match the tracking feature points of $\boldsymbol{f}_b$ with the corresponding feature points of $\bar{\boldsymbol{f}}$ using the feature point tracker directly, a great number of tracking feature points could be lost in case of large rotation, scale change or translation of the image, since a feature point tracker assumes only a small movement. To deal with these geometric transformations of the pixels of $\boldsymbol{f}_b$ to $\bar{\boldsymbol{f}}$, we use the SIFT features to estimate a homography [5] to retrieve a close position of the tracked feature points in image $\bar{\boldsymbol{f}}$ as an initial guess. For this purpose, we calculate a set of the coordinates of the N_B best matching SIFT features in image $\boldsymbol{f}_b$ and $\bar{\boldsymbol{f}}$

$$\boldsymbol{M} = \{(\boldsymbol{x}_1, \boldsymbol{x}'_1), (\boldsymbol{x}_2, \boldsymbol{x}'_2), \ldots, (\boldsymbol{x}_{N_B}, \boldsymbol{x}'_{N_B})\}$$

where $\boldsymbol{x} = (x, y, 1)^T$. We assume that the transformation of $\boldsymbol{x}_i$ to $\boldsymbol{x}'_i$ is a perspective transformation

$$\boldsymbol{M}_2 = \boldsymbol{H}\boldsymbol{M}_1 \qquad (5)$$

where matrix $\boldsymbol{H} \in \mathbb{R}^{3\times3}$ is a homography matrix and

$$\begin{aligned} \boldsymbol{M}_1 &= (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{N_B}) \\ \boldsymbol{M}_2 &= (\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_{N_B}). \end{aligned}$$

As $N_B > 3$, we use a least square estimation, namely the pseudo-inverse [17], to solve this overdetermined linear equation system of Equ. (5).

To estimate the positions of the tracked feature points in $\bar{\boldsymbol{f}}$, we use $\boldsymbol{H}$ to map the 2-D positions of the tracked feature points from $\boldsymbol{f}_b$ to $\bar{\boldsymbol{f}}$ as illustrated in Fig. 2.

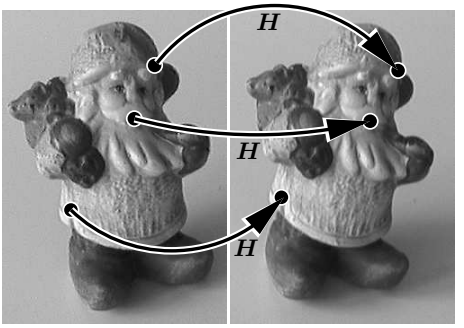

Figure 2: Illustration of the translation of the tracked feature points (marked as circles) using the homography matrix $\boldsymbol{H}$

4 Rendering feedback

So far, the camera parameter estimation has only used the original camera images, which the light field is based upon, for image matching. One of the

purposes of light fields, however, is to render photo-realistic images of the acquired scene from arbitrary view points.

For any camera parameters $\boldsymbol{v}$, the light field can be used to generate the image $\boldsymbol{f}_{\boldsymbol{v}}$ corresponding to those parameters. $\boldsymbol{v}$ contains the extrinsic camera parameters, i. e. position and orientation of a camera. We can use this generated image to search for the camera parameters $\bar{\boldsymbol{v}}$ of the new image $\bar{\boldsymbol{f}}$ by searching for the $\boldsymbol{v}$ such that $\boldsymbol{f}_{\boldsymbol{v}}$ is *closest* to $\bar{\boldsymbol{f}}$. We call this approach a *rendering feedback* approach, since we use images from the light field itself for the purpose of extending it.

Using a distance metric to compare images, the parameter search becomes a global minimization problem. For this work, we used a sum-of-squared-differences (SSD) distance metric. SSD is very efficient and can be used with multi-channel (color) images. SSD's main drawback is its lighting dependence, however to extend a light field the new images must necessarily be recorded with the same lighting conditions.

Several optimization algorithms can be applied to this problem. We have adopted two (related) approaches: one using adaptive random search (ARS) [16], and one using a particle filter (PF), specifically the Condensation algorithm [8].

Both approaches maintain a current set $\boldsymbol{V}_t$ of camera parameter *hypotheses* $\boldsymbol{v}_{t,i}, i = 1, \ldots, |\boldsymbol{V}_t|$, where t is an iteration index and $|\boldsymbol{V}_t|$ the fixed size of the set. Each hypothesis $\boldsymbol{v}_{t,i}$ also has a scalar rating $\nu_{t,i}$, derived from the image comparison. The initial set $\boldsymbol{V}_0$ can be derived from the camera parameters of the first sequence for a global search, or clustered around an initial estimate. Both approaches iterate over this set several times to fine-tune it, by generating new hypotheses $\boldsymbol{v}_{t+1,i}$ and ratings $\nu_{t+1,i}$ from the current ones. The methods differ in how the new hypothesis set is generated.

For the ARS, the rating is unchanged from the SSD distance measure. $\boldsymbol{V}_{t+1}$ is generated from $\boldsymbol{V}_t$ by discarding the worse rated half of all $\boldsymbol{v}_{t,i}$, and replacing them with diffused copies of the better rated half. The diffusion is an additive Gaussian noise, the standard deviation of which is derived from the spread of the original camera parameters. This standard deviation is reduced on each iteration to decrease the search area.

The PF has been used in conjunction with light fields for a model-based object tracking implementation [19]. Unlike the adaptive random search, the PF algorithm uses a probabilistic framework.

The rated hypothesis set represents the probability density function (pdf) of the camera position, $p(\boldsymbol{v}_t|\bar{\boldsymbol{f}})$, given the target image. Such rated hypotheses are also called *particles*. With Bayes' formula, we get

$$p(\boldsymbol{v}_t|\bar{\boldsymbol{f}}) = \frac{1}{c} p(\bar{\boldsymbol{f}}|\boldsymbol{v}_t) p(\boldsymbol{v}_t) \tag{6}$$

with c a normalizing constant. Thus, we seek information about the camera parameters $\boldsymbol{v}_t$ corresponding to the new image $\bar{\boldsymbol{f}}$. As with the ARS, the initial a priori density $p(\boldsymbol{v}_0)$, represented by $\boldsymbol{V}_0$ can be uniformly distributed over the search space or clustered around an initial estimate $\boldsymbol{v}_0$.

The a priori density $p(\boldsymbol{v}_t)$ is derived from the previous a posteriori density $p(\boldsymbol{v}_{t-1}|\bar{\boldsymbol{f}})$ through

$$p(\boldsymbol{v}_t) = \int p(\boldsymbol{v}_t|\boldsymbol{v}_{t-1}) p(\boldsymbol{v}_{t-1}|\bar{\boldsymbol{f}})\, d\boldsymbol{v}_{t-1} \quad . \tag{7}$$

In typical particle filter usage, this models the noisy state transition over time. Since the state is not expected to change, this is merely a diffusing process, as with the ARS.

The *likelihood* $p(\bar{\boldsymbol{f}}|\boldsymbol{v}_t)$ is derived from the image comparison by constructing a Gibbs distribution [1]:

$$p(\bar{\boldsymbol{f}}|\boldsymbol{v}_t) = \frac{1}{z} \exp\left(-\lambda E(\bar{\boldsymbol{f}}|\boldsymbol{v}_t)\right) \tag{8}$$

with z a normalizing constant. For each hypothesis $\boldsymbol{v}_{t,i}$, the rating is $\nu_{t,i} = p(\bar{\boldsymbol{f}}|\boldsymbol{v}_{t,i})$. The term $E(\bar{\boldsymbol{f}}|\boldsymbol{v}_t)$ is an error energy, comparing the target image with the image corresponding to the hypothesis $\boldsymbol{v}_t$. The better the hypothesis image matches the target, the lower the energy should be. Our image comparison metric, the SSD over the image space, has such a property, and is used unchanged for $E(\bar{\boldsymbol{f}}|\boldsymbol{v}_t)$. However, this may result in very similar energies for all compared images, which erodes the particle filter's effectiveness. Multiplying the SSD by a scalar value $\lambda > 1$ requires a hypothesis to be much closer for a good rating.

Using this likelihood definition as a hypothesis rating, the PF solves the combination of equations (6) through (8) using Monte Carlo integration. A new set $\boldsymbol{V}_{t+1}$ is derived from $\boldsymbol{V}_t$ by sampling from the latter, using the ratings as a sampling probability, and then re-rating $\boldsymbol{V}_{t+1}$. This new set represents the pdf whose main mode is the hypothesis $\bar{\boldsymbol{v}}$ with the lowest image discrepancy from $\bar{\boldsymbol{f}}$.

	office	*santa*	*candy*
feat. dist (no $\boldsymbol{H}$)	57.8	52.0	121.0
feat. dist ($\boldsymbol{H}$)	5.85	25.7	6.27
% tracked (no $\boldsymbol{H}$)	38.7%	64.4%	35.3%
% tracked ($\boldsymbol{H}$)	67.2%	70.1%	68.6%

Table 1: Tracking accuracy of SIFT features in pixels and percentage of features tracked without (no $\boldsymbol{H}$) and with ($\boldsymbol{H}$) using homography matrix for prediction

	office	*santa*	*candy*
ARS rot. α	0.800°	10.891°	1.035°
ARS rot. β	0.634°	4.854°	0.556°
ARS rot. γ	0.595°	9.291°	0.699°
ARS trans.	105.9%	799.9%	152.2%
PF rot. α	2.579°	4.717°	9.688°
PF rot. β	1.600°	7.626°	6.281°
PF rot. γ	0.986°	10.082°	5.284°
PF trans.	299.2%	712.4%	1287.7%

Table 2: Correctness of estimation of closest image for adaptive random search and particle filter

There are two caveats with this method. If the optimization starts from a single initial state estimate, the standard deviation of the particle diffusion must be chosen large enough so that the particles search beyond any local minima. Secondly, the results may by biased due to the rendering of light field images. Due to the rendering methods, an image rendered from a light field will usually display moderate to strong local distortion. This naturally causes the minimum to diverge slightly from the true camera parameters of $\bar{\boldsymbol{f}}$.

5 Experiments

The methods introduced before were tested on three different image sequences from a hand-held camera, *office* (109 images), *santa* (207 images) and *candy* (113 images). For each of these scenes, a second sequence was recorded starting at an arbitrary camera position within, or close to, the convex hull of the camera positions of the first sequence.

Two sets of experiments were performed to test the SIFT feature image matching[1]. For the first set, only the first image sequence was used. One of the images was removed from the sequence and then given to the search algorithm to perform a search for the nearest image. This was done for 10 images for each of the three scenes. The SIFT feature method found an image neighboring the missing image in the sequence in 100% of the experiments.

The second set of experiments dealt with attaching a second sequence to the original sequence. Again, 10 images were used, this time the first 10 from the second sequence. The SIFT feature neighbor detection then calculated the index of the image from the first sequence nearest to the target image. The homography $\boldsymbol{H}$ was then determined as in Sect. 3.3. The reconstruction of the camera poses of the first sequence was then extended by the target image. The nearest neighboring image was used as a starting point for feature tracking.

The extension was evaluated by measuring the average feature distance from the expected position, and the fraction of successfully matched tracking features with and without the homography. Table 1 shows the results. It is obvious that using the homography matrix improves the number of features tracked and the feature distance, in some cases dramatically. The homography allows the feature search to start much closer to the actual location.

For the *santa* scene, on a Pentium IV 2.4 GHz processor, a typical neighbor detection took about 120 seconds. Calculating the homography $\boldsymbol{H}$ took an additional 4 seconds.

To test the rendering feedback, two sets of experiments were again performed, similar to the SIFT feature tests. The first set again deals with finding the parameters of missing images, using the same setup as above for a global search. The result of the search, i. e. the proposed camera parameters of the removed image, were then compared to the original, ground-truth camera parameters from the image sequence including the missing image, calibrated as in Sect. 2.1. This was done for 10 images for each scene for both optimization methods. Both methods used the same number of particles and iterations.

Table 2 shows the average rotational and translational error. The rotational error is given as cardan angles in absolute degrees. Since the light fields use arbitrary coordinate units, the translational error for removed image $\boldsymbol{f}_i$ is calculated as $|\bar{\boldsymbol{t}} - \boldsymbol{t}_i| / |\boldsymbol{t}_{i+1} - \boldsymbol{t}_i|$, where $\bar{\boldsymbol{t}}$ is the found translation and $\boldsymbol{t}_i$ the calibrated translation of image $\boldsymbol{f}_i$. Thus, the translational error is given as a percentage of the average camera distance around the removed image.

[1] For detection and calculation of the SIFT features the software of D. Lowe was used, which can be downloaded at http://www.cs.ubc.ca/~lowe/keypoints/.

scene	*office*		*santa*		*candy*	
algorithm	ARS	PF	ARS	PF	ARS	PF
feature dist (all)	14.0	48.9	22.5	23.4	35.9	57.8
feature dist (init)	13.3	15.1	35.2	32.0	28.1	64.2
% tracked (all)	68.8%	68.1%	62.6%	62.3%	63.4%	54.8%
% tracked (init)	69.0%	68.7%	61.6%	60.9%	64.9%	55.9%

Table 3: Tracking accuracy of adaptive random search and particle filter in pixels and percentage of features tracked for a full search (all) or initialized with the closest image (init)

As can be seen, the ARS method outperforms the PF approach in two out of three sequences. The translational distances are generally within a few neighboring images, and the rotational ones within a few degrees. Though this error is larger than if the closest image as per the SIFT method had been used, the results are quite usable for a global search, especially since the rendered images are highly dependant on accurate depth maps, which were not always available. The number of particles in the *santa* scene, equaling the number of camera images, is larger than in the other scenes. Since both methods use the same image comparison, it is expected that with more iterations or particles, the PF method will match the ARS. However, for a limited number of iterations, the ARS converges faster.

For the *santa* scene, on a Pentium IV 2.66 GHz processor, 20 iterations at 207 particles take about 20 minutes for ARS, and 35 minutes for PF, mostly due to the time-intensive rendering of the images.

The second set of feedback experiments again dealt with attaching a second sequence. Initializing of the search was tested both from all first sequence camera parameters, and from the closest camera as per the SIFT neighbor search. The resulting proposed camera parameters were then passed to the light field extension process by using the projections of every known 3-D point as an estimate for the feature positions similar to the homography $\boldsymbol{H}$.

The results are listed in table 3. The slower convergence of PF in the *office* and *candy* scenes is reflected in the larger feature distance. However, initialization from a closest image is often beneficial for all approaches. The percentage of tracked features is comparable for all methods and situations.

Figure 3 finally shows the result of adding the second *santa* sequence to the first one. Image (a) shows the reconstruction already seen in Fig. 1 but augmented by the camera positions of the additional sequence. Images (b) and (c) show two images rendered from the resulting light field without (b) and with (c) the additional images, demonstrating the increased viewing range of the extended light field as well as a reduction in distortions.

6 Conclusion

We have presented two enhancing methods for solving the problem of accurately adding image data to a light field from an image sequence taken with a hand-held camera. The first method based on SIFT features significantly improves the point tracking from the original image sequence to the additional images. The second method obtains an estimate for the camera pose of a new image by using images rendered from the light field as state hypotheses in a parameter search.

The experiments have shown that using one or both methods successively reliably solves the problem of extending a light field. Nevertheless, both methods may yet be improved, the SIFT feature approach e. g. by taking into account clusters of votes in neighboring images, and the pose estimation will benefit from any improvement in rendering quality.

References

[1] A. Blake and A. Yuille, editors. *Active Vision*. MIT Press, Cambridge, Massachusetts, London, England, 1992.

[2] C. Buehler, M. Bosse, L. McMillan, S. J. Gortler, and M. F. Cohen. Unstructured lumigraph rendering. In *Proceedings of SIGGRAPH '01*, pages 425–432, Los Angeles, August 2001. ACM Press.

[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] R. Hartley. Euclidean reconstruction from uncalibrated views. In *Applications of Invari-*

(a) (b) (c)

Figure 3: Extended reconstruction of the *santa* image sequence (a) and rendered images without (b) and with (c) the additional input images as seen from exactly the same camera position

ance in Computer Vision, volume 825 of *Lecture Notes in Computer Science*, pages 237–256. Springer, Berlin, Heidelberg, New York, 1994.

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

[6] B. Heigl. *Plenoptic Scene Modeling from Uncalibrated Image Sequences*. ibidem-Verlag Stuttgart, January 2004.

[7] 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. Springer, Berlin, Heidelberg, New York, September 1999.

[8] M. Isard and A. Blake. Condensation — conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

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

[10] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th International Conference on Computer Vision*, volume 2, pages 1150–1157, Corfu, 1999. IEEE Computer Society, Washington.

[11] K. Mikolajczk and C. Schmid. A performance evaluation of local descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 257–263, Madison, 2003.

[12] 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.

[13] H. Schirmacher, C. Vogelgsang, H.-P. Seidel, and G. Greiner. Efficient free form light field rendering. In *Workshop Vision, Modeling and Visualization*, pages 249–256,528, Saarbrücken, Germany, Nov. 2001. Infix, Sankt Augustin.

[14] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, April 1991.

[15] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[16] A. Törn and A. Žilinskas. Global Optimization. *Lecture Notes in Comuter Science*, 3350, 1989.

[17] E. Trucco and A. Verri. *Introductory Techniques for 3–D Computer Vision*. Prentice Hall, New York, 1998.

[18] T. Zinßer, C. Gräßl, and H. Niemann. Efficient feature tracking for long video sequences. In *Pattern Recognition, Proceedings of 26th DAGM Symposium*. Springer-Verlag, Berlin, Heidelberg, New York, August 2004. To appear.

[19] M. Zobel, M. Fritz, and I. Scholz. Object tracking and pose estimation using light-field object models. In *Vision, Modeling, and Visualization 2002*, pages 371–378, Erlangen, Germany, Nov. 2002. Infix, Sankt Augustin.