Feature Management for Efficient Camera Tracking

Harald Wuest[1,2], Alain Pagani[2], Didier Stricker[2]

[1] Centre for Advanced Media Technology (CAMTech)
Nanyang Technological University (NTU)
50 Nanyang Avenue, Singapore 649812
[2] Department of Virtual and Augmented Reality
Fraunhofer IGD
TU Darmstadt, GRIS, Germany
Harald.Wuest@igd.fraunhofer.de

Abstract. In dynamic scenes with occluding objects many features need to be tracked for a robust real-time camera pose estimation. An open problem is that tracking too many features has a negative effect on the real-time capability of a tracking approach. This paper proposes a method for the feature management, which performs a statistical analysis of the ability to track a feature and then uses only those features which are very likely to be tracked from a current camera position. Thereby a large set of features in different scales is created, where every feature holds a probability distribution of camera positions from which the feature can be tracked successfully. As only the feature points with the highest probability are used in the tracking step, the method can handle a large amount of features in different scale without losing the ability of real time performance. Both the statistical analysis and the reconstruction of the features' 3D coordinates are performed online during the tracking and no preprocessing step is needed.

1 Introduction

Tracking point based features is a widely used technique for the camera pose estimation. Either reference features are taken from pre-calibrated images with a given 3D model [1, 2] or the feature points are reconstructed online during the tracking [3–5]. These approaches are very promising, if the feature points are located on well textured planar regions. However, in industrial scenarios objects often consist of reflecting materials and poorly textured surfaces. Because of spotlights or occluding objects, the area of camera positions where a feature point has the same visual appearance can be very limited. Increasing the number of features can help to ensure a robust camera pose estimation, but as the 2D feature tracking step makes up a big amount of the computation time, the overall tracking performance gets very poor. Using only a subset of those features, which are visible from a given view point, can avoid this problem.

Najafi et al. [1] present a statistical analysis of the appearance and shape of features from possible viewpoints. In an offline training phase they coarsely

sample the viewing space at discrete camera positions and create cluster groups of viewpoints for every model feature according to similar feature descriptors. Thereby a map is created which gives information about the detection repeatability, accuracy and visibility from different viewpoints for every feature. During the online phase this information is used for a selection of good features. In this paper we present a method for a feature management which does not rely on any preprocessing but performs an online estimation of the tracking probability of every feature. The ability to track a feature is observed during the runtime and a distribution of camera positions of tracking successes and tracking failures is created. These distributions are represented by a mixture model with a constant number of Gaussians. A merge operation is used to keep the number of Gaussians fixed. The resulting tracking probability, which not only models the visibility but also the robustness of a feature, is then used to decide which features are most suitable to be tracked at a given camera position. The robust camera pose estimation is solved by using Levenberg-Marquardt minimization and RANSAC outlier rejection.

2 Feature Tracking and Reconstruction

For a robust reconstruction and pose estimation a feature point must be tracked as long as possible. Therefore it should be invariant to deformations, illumination and scale. The well-known Shi-Tomasi-Kanade tracker is a widely used techniques for tracking 2D feature points [6]. It is based on the iterative minimization of the sum of squared differences with a gradient decent method. In [7] illumination compensation has been added in the minimization procedure. The problem of updating a template patch has been addressed in [8]. Another promising approach for a reliable 2D feature tracking was presented by Zinßer et al.[9], where a brightness corrected affine warped template patch is used to track a feature point. They proposed a two-stage approach, where pure translation from frame to frame is estimated first on several levels of the image pyramid, an then the template patch is iteratively aligned at the resulting image position of the first stage. The alignment of the patch T in the image I is based on minimizing the following squared intensity difference

$$\epsilon = \sum (I(\boldsymbol{x}) - (\lambda T(g_{\boldsymbol{\alpha}}(\boldsymbol{x})) + \delta))^2, \tag{1}$$

where λ and δ are the parameters for adjusting the contrast and the brightness, and $g_{\boldsymbol{\alpha}}$ is the affine transformation function. We extended this method by extracting a template patch in different resolution levels of the image pyramid and always select that patch, which has the most similar resolution to the predicted affine transformed patch. If the desired resolution of the patch does not exist, it is extracted out of the current image after a successful tracking step. A feature is regarded as tracked successfully, if the iterations of the alignment converge and the error of equation 1 is smaller than a given threshold.

Successfully tracked features are reconstructed by triangulation and further refined by an Extended Kalman Filter. More details can be found in [5].

3 Feature Management

The functions of the feature management are the extraction of new features, the estimation of the feature tracking probability, the selection of good features for a given camera position and the removal of features which are not of any use for further tracking. The whole management shall be an incremental process which runs in real-time and only uses a limited amount of memory. The tracking probability of a feature is denoted as the probability if a feature is able to be tracked successfully at a given camera position. In the following section the sequential estimation of this probability is described.

3.1 Tracking Probability

As the rotation around the camera center does not have any influence on the visibility of a point feature, if the feature is located inside the image, only the position of the camera in world coordinates is regarded as useful information to decide, whether a feature is worth tracking. What is known about the ability to track a feature at a given camera position are the observations of its tracking success in previous frames. The problem of modeling a probability distribution $p(\boldsymbol{x})$ of a random variable $\boldsymbol{x}$, given a finite set $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ of observations, is known as density estimation. A widely used nonparametric method for creating probability distributions are Kernel density estimators. To obtain a smooth density model we choose a Gaussian kernel function. For a D-dimensional vector $\boldsymbol{x}$ the probability density can be denoted as

$$p(\boldsymbol{x}) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_n\|^2}{2\sigma^2}\right) \tag{2}$$

where N is the number of observation points $\boldsymbol{x}_n$, and σ represents the variance of the Gaussian kernel function in one dimension.

Every observation of a feature belongs to one element of the class $C = \{s, f\}$, which simply holds the information whether the tracking step was successful (s) or the tracking failed (f). The probability density of the camera position is estimated for every element of the class C separately. Let $p(\boldsymbol{x}|C = s)$ be the conditional probability density of the camera position for successfully tracked features and $p(\boldsymbol{x}|C = f)$ the conditional probability density for unsuccessfully tracked features. The marginal probability of tracking successes is given by $p(C = s) = \frac{N_s}{N}$ and for tracking failures by $p(C = f) = \frac{N_f}{N}$, where N_s and N_f are the number of successful and unsuccessful tracking steps respectively, and N is the total number of observations.

The probability $p_t(\boldsymbol{x})$ if a feature can be tracked at a given camera position $\boldsymbol{x}$ is estimated with

$$p_t(\boldsymbol{x}) = p(C = s|\boldsymbol{x}) \tag{3}$$

When applying the Bayes' theorem, the tracking probability can be written as

$$\begin{aligned} p_t(\boldsymbol{x}) &= \frac{p(\boldsymbol{x}|C=s)p(C=s)}{p(\boldsymbol{x})} \\ &= \frac{p(\boldsymbol{x}|C=s)p(C=s)}{p(\boldsymbol{x}|C=s)p(C=s)+p(\boldsymbol{x}|C=f)p(C=f)} \\ &= \frac{p(\boldsymbol{x}|C=s)N_s}{p(\boldsymbol{x}|C=s)N_s+p(\boldsymbol{x}|C=f)N_f} \end{aligned} \tag{4}$$

The estimation of probability densities by using equation 2, however, has the major drawback that with an increasing number of observations the complexity for storage and computation is increasing linearly with the number of observations, which is not feasible for an online application. Our approach for the density estimation is based on a finite set of Gaussian mixtures.

The use of mixture models for an efficient computation of clusters in huge data sets has already been addressed. In [10] the Iterative Pairwise Replacement Algorithm (IPRA) is proposed, which is a computational efficient method for conditional density estimation for very large data sets, where kernel estimates are approximated by much smaller mixtures. Goldberger [11] uses an hierarchical approach to reduce large Gaussian mixtures to smaller mixtures by minimizing a KL-based distance between them. Zhang[12] present another efficient approach for simplifying mixture models by using a L_2 norm as distance measure between the mixtures. Zivkovic [13] presents a recursive solution for estimating the parameters of a mixture with a simultaneous selection of the number of components.

We use a method which is similar to [10], but instead of clustering a large data set, we use the method for an online density estimation with a finite mixture model. A mixture with a finite number of Gaussians is maintained for both the successfully and unsuccessfully tracked features. Now we regard the multivariate Gaussian mixture distribution of the successfully tracked features, which can be written as

$$p(\boldsymbol{x}|C=s) = \sum_{k=1}^{K} \omega_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \Sigma_k) \quad \text{with} \quad \sum_{k=1}^{K} \omega_k = 1 \tag{5}$$

where $\boldsymbol{\mu}_k$ is the D-dimensional mean vector and Σ_k the $D \times D$ covariance matrix. The mixing coefficients $\omega_k = \frac{N_k}{N_s}$ hold the information how many observations N_k have affected this Gaussian k. The probability distribution $p(\boldsymbol{x}|C=f)$ is defined in the same way. Together with equation 4 the tracking probability for a given camera position can be estimated.

The mixture model is built and maintained as follows. Depending on the tracking success, an observation is assigned to a class C, which means that either the distribution $p(\boldsymbol{x}|C=s)$ or the distribution $p(\boldsymbol{x}|C=f)$ is updated. First for every observation a Gaussian kernel function is created, where every kernel can be regarded as a Gaussian of the mixture model. If the maximum number of

mixtures K is reached, then the two most similar mixtures are merged and a new Gaussian is created by taking the kernel function from the proximate observation.

3.2 Similarity Measure

A similarity matrix is maintained, where the similarity of all Gaussians among each other is stored.

Scott [10] defined the similarity measure between two density functions p_1 and p_2 as

$$\mathrm{sim}(p_1, p_2) = \frac{\int_{-\infty}^{\infty} p_1(x)p_2(x)dx}{(\int_{-\infty}^{\infty} p_1^2(x)dx \int_{-\infty}^{\infty} p_2^2 dx)^{1/2}} \tag{6}$$

Equation 6 can be considered as a correlation between the two densities.

When $p_1(x) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_1, \Sigma_1)$ and $p_2(x) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_2, \Sigma_2)$ are normal distributions, then the similarity measure can be calculated by

$$\mathrm{sim}(p_1, p_2) = \frac{(2^D|\Sigma_1\Sigma_2|^{1/2})^{1/2}}{|\Sigma_1 + \Sigma_2|^{1/2}} \exp(\Delta) \tag{7}$$

with

$$\Delta = -\frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T(\Sigma_1 + \Sigma_2)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2). \tag{8}$$

This equation follows from the fact that

$$\int_{-\infty}^{\infty} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_2, \Sigma_2) = \mathcal{N}(\mathbf{0}|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2, \Sigma_1 + \Sigma_2). \tag{9}$$

The two Gaussians for which the similarity measure of equation 6 is smallest, are used for the merging step, which is described in the next section.

3.3 Merging Gaussian Distributions

The merge operation of the two most similar Gaussians is carried out as follows. Now we assume that the i^{th} and the j^{th} component are merged into the i'^{th} component of the mixture. Since a mixing coefficient represents the number of observations, which affect a distribution, the new number of observations is $N_{i'} = N_i + N_j$, and therefore $\omega_{i'}$ is updated by

$$\omega_{i'} = \omega_i + \omega_j. \tag{10}$$

The mean of the new distribution can be calculated by

$$\begin{aligned} \boldsymbol{\mu}_{i'} &= \frac{1}{N_{i'}} \sum_{n=1}^{N_{i'}} \boldsymbol{x}_n = \frac{1}{N_{i'}} (\sum_{n=1}^{N_i} \boldsymbol{x}_n + \sum_{n=1}^{N_j} \boldsymbol{x}_n) \\ &= \frac{1}{N_{i'}} (N_i\boldsymbol{\mu}_i + N_j\boldsymbol{\mu}_j) = \frac{1}{\omega_{i'}} (\omega_i\boldsymbol{\mu}_i + \omega_j\boldsymbol{\mu}_j) \end{aligned} \tag{11}$$

After the mean is computed, the covariance $\Sigma_{i'}$ can be updated as follows

$$\begin{aligned}
\Sigma_{i'} &= \frac{1}{N_{i'}} \sum_{n=1}^{N_{i'}} (\boldsymbol{x}_n - \boldsymbol{\mu}_{i'})(\boldsymbol{x}_n - \boldsymbol{\mu}_{i'})^T \\
&= \frac{1}{N_{i'}} \sum_{n=1}^{N_{i'}} \boldsymbol{x}_n \boldsymbol{x}_n^T - \boldsymbol{\mu}_{i'} \boldsymbol{\mu}_{i'}^T \\
&= \frac{1}{N_{i'}} (\sum_{n=1}^{N_i} \boldsymbol{x}_n \boldsymbol{x}_n^T + \sum_{n=1}^{N_j} \boldsymbol{x}_n \boldsymbol{x}_n^T) - \boldsymbol{\mu}_{i'} \boldsymbol{\mu}_{i'}^T \\
&= \frac{1}{N_{i'}} (N_i(\Sigma_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T) + N_j(\Sigma_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^T)) - \boldsymbol{\mu}_{i'} \boldsymbol{\mu}_{i'}^T \\
&= \frac{1}{\omega_{i'}} (\omega_i(\Sigma_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T) + \omega_j(\Sigma_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^T)) - \boldsymbol{\mu}_{i'} \boldsymbol{\mu}_{i'}^T. \qquad (12)
\end{aligned}$$

After the merge operation, the j^{th} component can be used by a new observation to represent a new Gaussian. It can be regarded as a Kernel estimate with a Gaussian kernel function. For a new observation, the camera position is assigned to $\boldsymbol{x}_j$ and the covariance is set to $\sigma^2 \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix and σ determines the size of the Parzen window. The parameter σ affects the smoothness of the resulting mixture model and must be chosen with respect to the world coordinate system. If for example the camera position vector is given in cm, with $\sigma = 5$ a convincing probability distribution can be created for an indoor camera tracking. The weight ω_j is initialized with $\omega_j = \frac{1}{N_c}$, where N_c is the number of observation of the assigned class.

3.4 Feature Selection

Features which have a precisely reconstructed 3D coordinate have no need for any reconstruction or refinement step. If we know, that such features are not very likely to be tracked from the current camera position, it is probably not of any use for the pose estimation and it can be disregarded for a tracking step. Features which do not have a valid 3D coordinate are selected for the tracking step in every case, because it is important, that a feature point gets triangulated fast, and an exact 3D position is reconstructed, so that the feature will be beneficial for the camera pose estimation.

Before the tracking step, all features which have not been tracked successfully in the last frame are projected into the image with the last camera position in order to provide a good starting position for the features in the iterative alignment. The tracking probabilities of all features which are located inside the current image are calculated with equation 4 and the features are sorted by their probability in descending order. Now the feature tracking described in section 2 is applied on the sorted list of features until a minimum number of features has been tracked successfully. In our implementation we stop after 30 successfully tracked features with a valid 3D coordinate, which should be enough for a robust pose estimation.

The benefit of this approach is that the total number of tracked features is kept at a minimum if most of the features are tracked successfully, but if there are lots of tracking failures due to occlusion or strong motion blur, as many features as needed are tracked until a robust camera pose estimation is possible.

3.5 Feature extraction

Most point based feature tracker use the well known Harris Corner Detector [14], which is based on the eigenvalue analysis of the gradient structure of an image patch. Another simple but very efficient approach called FAST (Features from Accelerated Segment Test) was presented by Roston et al.[15]. Their method analyses the intensity values on a circle of 16 pixels surrounding the corner point. If at least 12 contiguous pixels are all above or all below the intensity of the center by some threshold, this point is regarded as a corner feature. For reasons of efficiency we used the FAST feature detector in our implementation.

To avoid too many features and overlapping patches, a new feature is only extracted, if no other feature points exist within a minimum distance to this feature in the image. New features are extracted if the total number of features with $p_t(\boldsymbol{x}) > 0.5$ for the current camera position $\boldsymbol{x}$ falls below a given threshold.

3.6 Feature removal

In order to decide, if a feature is valuable for further tracking, a measure of usefulness has to be defined. If the tracking probability $p_t(\boldsymbol{x})$ for any camera position x is smaller than 0.5, a feature can be regarded as dispensable. The correct computation of the maximum of $p_t(\boldsymbol{x})$ with the expectation maximization algorithm for every feature is computationally too expensive.

When $\boldsymbol{\mu}_{k,s}$ are the Gaussian means of the mixture model representing successfully tracked features, we approximate the maximum of the tracking probability by evaluating p_t at all positions $\boldsymbol{\mu}_{k,s}$ by the following equation:

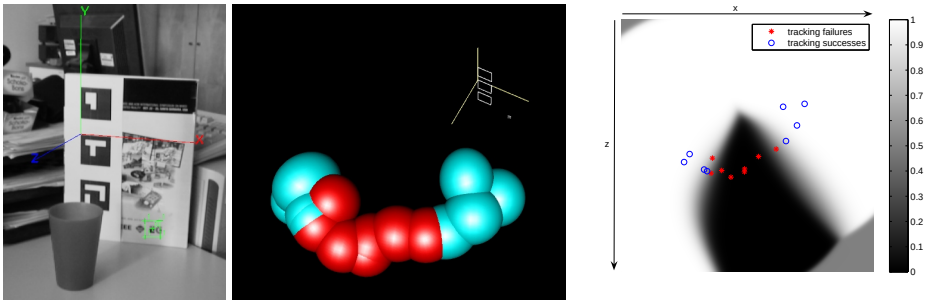
$$p_{\max} \simeq \max_k p_t(\boldsymbol{\mu}_{k,s}) \tag{13}$$

If $p_{\max} < 0.5$ holds, then no camera position exists, where this feature is likely to be tracked, and it can be removed from the feature map without the concern of losing valuable information.

If a feature point gets lost and the 3D coordinate of that feature has not been reconstructed yet, this feature is removed as well, because without a valid 3D coordinate it is not possible to re-project the feature back into the image for further tracking.

4 Experimental Results

To evaluate if the tracking probability distribution of a single feature is estimated correctly the following test scenario is created. The camera pose is computed by



(a) (b) (c)

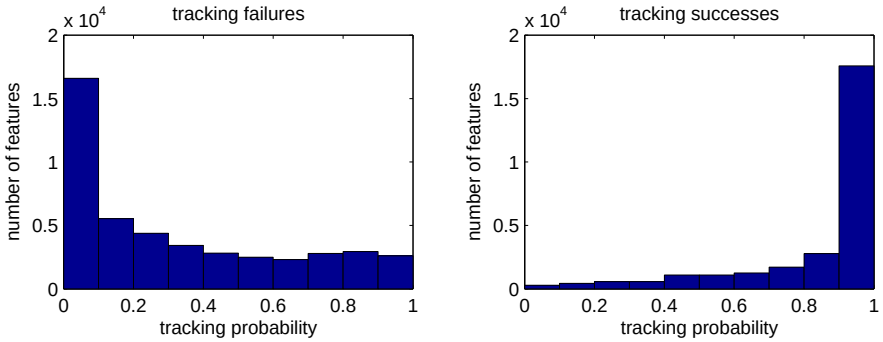
Fig. 1. Probability density map of camera position for a single feature. In (a) a frame of the test sequence is shown. (b) visualizes the Gaussian mixture models of camera positions, where the feature has been tracked successfully (blue) and where the tracking failed (red). In (c) the tracking probability in the x/z-plane can be seen.

tracking a set of planar fiducial markers, which are located in the x/y-plane. A point feature is extracted manually on the same plane. In figure 1(a) a frame of this sequence can be seen. When the camera is moved around, the point feature gets lost while it is occluded by an object, but it is tracked successfully, when it gets visible again. The Gaussian mixture model is visualized in figure 1(b) by a set of confidence ellipsoids, which are drawn in blue and red for $p(\boldsymbol{x}|C = s)$ and $p(\boldsymbol{x}|C = f)$ respectively. The number of Gaussians is limited to 8 for each mixture model in this particular example. In figure 1(c) the probability distribution $p_t(\boldsymbol{x})$ in the x/z-plane together with the Gaussian means is shown. It can be seen that the camera positions where the point feature was visible or occluded is correctly represented by the mixture model of tracking successes or tracking failures respectively. The probability distribution clearly illustrates that the tracking probability falls to 0 at camera positions where the feature is occluded.

An image sequence showing an industrial scenario is used for the further experiments. In order to evaluate the quality of the tracking probability estimation, all available features are used as input for the tracking step and it is observed whether the features compared to their tracking probability are tracked successfully or not. In figure 2 histograms are plotted, which show the number of successfully and unsuccessfully tracked features with their corresponding tracking probability. It can be seen that the major part of features with a high tracking probability has been indeed tracked successfully.

An analysis of the processing time is carried out on a Pentium 4 with 2.8GHz and a firewire camera with a resolution of 640×480 pixels. The average computational costs for every individual step are shown in table 1. Without the feature extraction, the tracking system can run at a frame rate of 20Hz.

If no feature selection is performed, in average 93.9 features are used in the feature tracking step, and only 49.0% of all features can be tracked successfully. The average runtime of the tracking step is at 64.36 milliseconds. With the



(a) (b)

Fig. 2. Histogram of successfully and unsuccessfully tracked features with their corresponding tracking probability.

selection of the most probable features in average only 48.94 features are analyzed per frame in the tracking step. The success rate of the feature tracking is at 83.0% and the mean computation time is lowered to 29.08ms with no significant difference of the quality of the pose estimation.

5 Conclusion

We have presented an approach for real-time camera pose estimation, which uses an efficient feature management to store many features and to track only those features, which are most likely to be tracked from a given camera position. The tracking probability for every feature is estimated online during the tracking and no preprocessing is necessary. Features which are only visible in a limited area of view points are only tracked at those certain camera positions and ignored at any other view points. Even if they are occluded for a long time, reliable features are not deleted, but kept in the feature set as long as a camera position exists, from where the feature can be tracked successfully. Not only the visibility, but also the robustness of a feature is represented by the tracking probability.

prediction step	**time in ms**
build image pyramid	10.53
feature selection and tracking	29.08
pose estimation	2.74
update feature probability	1.94
reconstruct feature points	5.53
extract new features	5.93
total time without feature extraction	49.82

Table 1. Average processing time of the individual steps of the tracking approach.

Tracking failures due to reflections or spotlights at certain camera positions are also modeled correctly.

Acknowledgements

This work was partially funded by the European Commission project SKILLS, Multimodal Interfaces for Capturing and Transfer of Skills (IST-035005, `www.skills-ip.eu`).

References

1. Najafi, H., Genc, Y., Navab, N.: Fusion of 3d and appearance models for fast object detection and pose estimation. In: Lecture Notes in Computer Science, 7th Asian Conference on Computer Vision (ACCV). (2006)
2. Bleser, G., Pastarmov, Y., Stricker, D.: Real-time 3d camera tracking for industrial augmented reality applications. In: WSCG (Full Papers). (2005) 47–54
3. Genc, Y., Riedel, S., Souvannavong, F., Akinlar, C., Navab, N.: Marker-less tracking for ar: A learning-based approach. In: IEEE / ACM International Symposium on Mixed and Augmented Reality. (2002) 295 – 304
4. Davison, A.: Real-time simultaneous localisation and mapping with a single camera. In: Proc. International Conference on Computer Vision, Nice. (2003)
5. Bleser, G., Wuest, H., Stricker, D.: Online camera pose estimation in partially known and dynamic scenes. In: ISMAR. (2006) 56–65
6. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94). (1994) 593 – 600
7. Jin, H., Favaro, P., Soatto, S.: Real-Time feature tracking and outlier rejection with changes in illumination. In: IEEE Intl. Conf. on Computer Vision. (2001) 684–689
8. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**(6) (2004) 810 – 815
9. Zinßer, T., Gräßl, C., Niemann, H.: Efficient Feature Tracking for Long Video Sequences. In: DAGM. (2004) 326–333
10. Scott, D.W., Szewczyk, W.F.: From kernels to mixtures. In: Technometrics. Volume 43. (2001) 323–335
11. Goldberger, J., Roweis, S.: Hierarchical clustering of a mixture model. In Saul, L.K., Weiss, Y., Bottou, L., eds.: Advances in Neural Information Processing Systems 17. MIT Press, Cambridge, MA (2005) 505–512
12. Zhang, K., Kwok, J.: Simplifying mixture models through function approximation. In Schölkopf, B., Platt, J., Hoffman, T., eds.: Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA (2007)
13. Zivkovic, Z., van der Heijden, F.: Recursive unsupervised learning of finite mixture models. IEEE Trans. Pattern Anal. Mach. Intell. **26**(5) (2004) 651–656
14. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proc. Alvey Vision Conf, Univ. Manchester (1988) 147–151
15. Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: IEEE International Conference on Computer Vision. Volume 2. (2005) 1508–1511