A Dataset and Evaluation Methodology for Template-based Tracking Algorithms

Sebastian Lieberknecht[*] metaio GmbH
Selim Benhimane[†] metaio GmbH
Peter Meier[‡] metaio GmbH
Nassir Navab[§] Technische Universität München

Abstract

Unlike dense stereo, optical flow or multi-view stereo, template-based tracking lacks benchmark datasets allowing a fair comparison between state-of-the-art algorithms. Until now, in order to evaluate objectively and quantitatively the performance and the robustness of template-based tracking algorithms, mainly synthetically generated image sequences were used. The evaluation is therefore often intrinsically biased.

In this paper, we describe the process we carried out to perform the acquisition of real scene image sequences with very precise and accurate ground truth poses using an industrial camera rigidly mounted on the end-effector of a high-precision robotic measurement arm. For the acquisition, we considered most of the critical parameters that influence the tracking results such as: the texture richness and the texture repeatability of the objects to be tracked, the camera motion and speed, and the changes of the object scale in the images and variations of the lighting conditions over time.

We designed an evaluation scheme for object detection and inter-frame tracking algorithms and used the image sequences to apply this scheme to several state-of-the-art algorithms. The image sequences will be made freely available for testing, submitting and evaluating new template-based tracking algorithms, i.e. algorithms that detect or track a planar object in an image sequence given only one image of the object (called the template).

1 Introduction

In the last few years, markerless visual tracking reached the level where a large variety of algorithms could be successfully used in a wide range of Augmented Reality applications [9, 20]. Until now, the performance of state-of-the-art algorithms was either evaluated quantitatively using synthetically rendered images or evaluated qualitatively using *ad hoc* recorded videos demoing a new method.

Using synthetic images has the advantage that the camera pose with respect to the synthetic scene can be set perfectly. On the one hand, comparing the pose estimation obtained using a given tracking algorithm with the camera pose set during the rendering process allows the establishment of the estimation error. On the other hand, it is very hard to create synthetic images that reproduce the real effects of every phenomenon such as lighting, noise, motion blur, discretization, blooming or limited color depths during the real image acquisitions. In general, these effects can only be approximated which makes the evaluations intrinsically biased. Such images could be used during the design of new algorithms and to get a rough impression of the behavior of the tracking but they do not guarantee the performances that will be obtained with real world data.

[*]e-mail:Sebastian.Lieberknecht@metaio.com
[†]e-mail:Selim.BenHimane@metaio.com
[‡]e-mail:Peter.Meier@metaio.com
[§]e-mail:Navab@cs.tum.edu

IEEE International Symposium on Mixed and Augmented Reality 2009
Science and Technology Proceedings
19 -22 October, Orlando, Florida, USA
978-1-4244-5389-4/09/$25.00 ©2009 IEEE

Using recorded videos of real scenes can, however, demonstrate that a given algorithm can or cannot work in real world conditions but the evaluation of the performances remains qualitative. For example, it makes it possible to visually see that the camera pose was or was not correctly estimated if the virtual augmentation is or is not positioned and oriented as expected.

Since the research community is working on markerless visual tracking very actively, the need of common objective datasets with ground truth is growing. We believe that image sequences acquired with a real camera where the pose of the camera for every frame of the sequences is known quite perfectly will give very strong benefits to the community. Such datasets will allow a fast performance estimation in terms of speed and accuracy of a newly designed algorithm and its fair comparison with the existing ones.

In this paper, we present the methodology we used to record image sequences with ground truth knowledge about the position and orientation of the camera in every frame. We used an industrial camera mounted on the end effector of a high-precision robotic measurement arm as hardware setup. The camera is calibrated intrinsically with an industrial 3D calibration target which is also used to compute the hand-eye-calibration, i.e. the transformation between the camera center and the measurement tip.

For this first version of the dataset, we recorded sequences of planar targets using the calibrated setup. We identified four different types of tracking targets classified by texture richness and repeatability. Each type is represented by two targets in the dataset. Next, we determined five standard factors that have the biggest influence on the performance of the tracking and which are related to the camera motion, the size of the tracked object in the image and the lighting conditions; one sequence per target is dedicated to each influence. The dataset has in total 40 sequences of 1200 images each.

The recorded sequences were first checked against very accurately detected corners of fiducials, the resulting average residual was below one pixel. We also used the dataset in order to evaluate four popular (and at least in compiled form available) state-of-the-art algorithms, three tracking-by-detection methods and one frame-by-frame tracking method. We summarize the evaluation results in the end of the paper.

2 Related Work

For a long time, Quam's Yosemite sequence [8] used to be the reference used for evaluating optical flow algorithms. Quam created a model by mapping aerial photos onto a depth map of the Yosemite valley and generated a sequence by simulating a flight through the valley.

Today, the Middlebury datasets [4] are the reference for optical flow. Besides having generated synthetic images, they also created dense ground truth by using hidden fluorescent texture. The same group additionally made ground truth datasets for dense stereo matching using structured light and datasets for multi-view stereo using a laser scanner [16, 17]. Theoretically, these images could be used to evaluate tracking algorithms as well. However, due to the very limited number of frames/image pairs given and the completely different goal set when creating these datasets, the result from an evaluation using these datasets will be missing important factors such as e.g. motion blur and the irregular movements coming from a human camera operator.

In markerless visual tracking, Baker and Matthews [3] used synthetically warped images to compare four different tracking algorithms, the warping amplitude and the noise of the image were simulated synthetically. Mikolajczyk and Schmidt [11] used still images for comparing affine region detectors, their dataset consisting of 48 images was also used by many others. Moreels and Perona [12] used a turn table together with a static stereo camera setup to evaluate the performance of feature detectors and descriptors on 3D objects. They generated a database consisting of 100 objects with calibrated views, one image pair for each 5° rotation of the turntable. In general, turn table sequences with a static camera only model a limited range of transformations wich are not specifically representative for AR applications.

Recently, Zimmerman, Matas and Svoboda [21] published a dataset consisting of three grayscale image sequences with transformations of the targets that cover all six degrees of freedom and that could be used to evaluate frame-by-frame tracking algorithms. The ground truth poses of these sequences were obtained by manually clicking on either crosses that were attached to objects or by clicking directly on the corners of an object. The three image sequences consist of approximately 12000 images in total and feature three different targets.

The problem is that this dataset only considers a very limited number of objects and factors influencing the tracking, e.g. the lighting conditions were kept fixed. Moreover, the ground truth data was based on information exclusively coming from the images, it was mainly done by clicking points in the images with pixel-accurate localization (no sub-pixel localization). Following this approach, it is not possible to have reliable ground truth in the case of blurry or noisy images. It is also not possible to recover the camera position and orientation when the points used to determine the pose are not in the field of view of the camera. Consequently, the performance of the tested algorithms could not be evaluated in the presence of noise, motion blur or for some relative position between the camera and the tracked objects.

We propose a novel dataset that methodically focusses on different types of tracking targets as well as on the factors with biggest influence on the tracking performance. The ground truth was established using dedicated hardware and was not exclusively dependent on the acquired image data.

3 Description of the setup

In the following, we describe the hardware components we use to obtain the ground truth pose data. In order to reach a very high precision, we use proven industrial hardware components wherever possible and appropriate.

A FaroArm Platinum [7], a robotic measurement arm, was used as key component for the ground truth measurements. It has seven axes and an accuracy better than 0.013 mm within its reach of around 1.2 m from its base. The arm uses internal temperature sensors to compensate the effect of thermal variations. We assume that the residual noise in the measurement arm is neglectibly small. An industrial AVT Marlin F-080C camera [2] was rigidly mounted on the end effector of the measurement arm. The camera is able to capture progressive VGA frames with 40 Hz while the measurement arm provides absolute pose data with 75 Hz. We used an adjustable panel light providing daylight-balanced 5600 Kelvin light for controlling the illumination. An image of the setup can be seen in figure 1.

The following steps were done to make the system operational: First, the intrinsics and undistortion coefficients of the camera were computed using a professional photogrammetric calibration target and the accompanying software from AICON [1]. The camera calibration was conducted following the guidelines of the provider, the

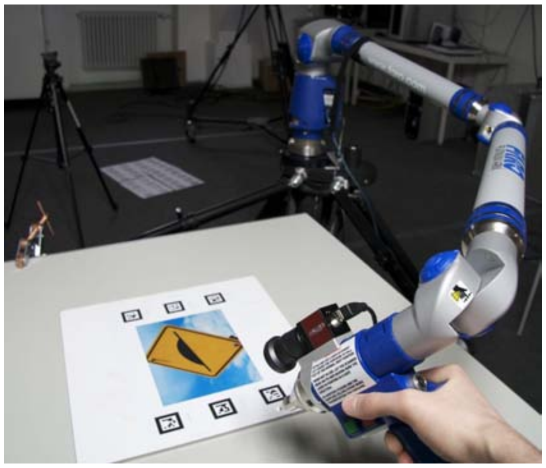


Figure 1: The measurement arm, the camera and one of the targets.

residual error was less than 0.07 pixels.

The intrinsics of the robot, namely its tip and the seven joints, were also calibrated according to the guidelines of the manufacturer. After that, the hand-eye-calibration was conducted. For this, we again used the AICON calibration target and software, but this time only used the extrinsic orientation together with poses of the faro arm. Ten pose pairs were recorded and the hand-eye-calibration was computed with a method similar to the one proposed by Tsai and Lenz [18].

4 Designing the dataset

The motivation for creating the dataset is to evaluate and to allow a fair comparison between a wide range of template-based tracking algorithms. There are algorithms that use corners, edges and whole regions of an image. We tried to make the dataset balanced and focussed on real world usage. In the following, we describe how we tried to achieve these goals.

4.1 Targets

The most basic, essential and quasi-incontrovertible task is the tracking of planar structures. In fact, in order to be able to determine the relative position and orientation of a camera with respect to different objects of the environment in real-time, most of the model-free detection and tracking algorithms have to assume that the objects are locally planar or piecewise planar. That is why we used planar targets. These targets were chosen to represent a broad overview of all types of possible tracking targets. We classified them into four different groups, namely "Low Texture", "High Texture", "Repetitive Texture" and "Normal Texture", meaning somewhere in between. Each class is represented by two targets each, shown in figure 2. We did not track any of these targets in advance in order to not make the selection of the targets biased.

The "Low Texture" group consists of images of road signs which are composed of two distinct colors and large uniform areas, thus large edges are visible. In the "Repetitive" group there are images of electronic boards, one image with mainly large, one image with mainly small components. An image of a car and of a cityscape are in the group of the "Normal" reference targets. Finally, the group of "Highly Textured" images is composed of an image of a wall made of different sized stones and of an image with English lawn which features many extremely small structures everywhere.

Each target image was resampled to a resolution of 800×600 before printing. The algorithms later were provided with the inner 640×480 area of these target images. We compared the results of the ground truth with the results of the tracking of six fiducials

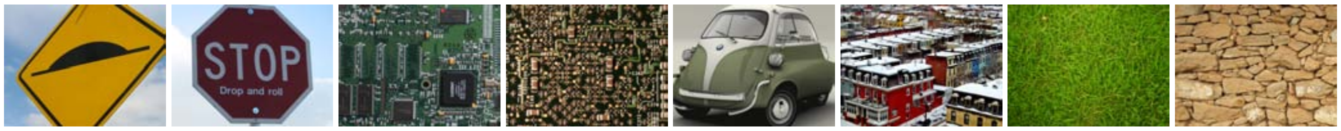


Figure 2: The reference targets used in our dataset. From left: Low, Repetitive, Normal, High Texturedness.

(markers) placed next to the reference targets (see figure 5) for final validations.

The targets were printed with a color laser printer. Both the printer and the camera were not specifically color calibrated such that the captured images of the printed targets match the reference targets exactly. These steps were skipped on purpose as they are common in real world scenarios and hard to emulate with synthetic images. The printed pages were glued onto foamboard which was later fixed on a table rigidly connected with the base of the measurement arm.

4.2 Sequences

Next, we designed the dynamic part of the evaluation. Here we focus on five different types of dynamic behaviors: “Angle”, “Range”, “Fast Far”, “Fast Close” and “Illumination”.

In the sequences of type “Angle”, we focus on varying the angle between the normal of the reference target and the optical axis of the camera between 0° and approximately 80° while trying to keep the distance to the target constant. The target covers around 10 - 30% of the image.

The “Range” sequences focus on the size of the reference image in the camera image. The maximum distance of the camera to the target resulted in a visible area of the reference target of about 130×100 pixels or 4% of the original area, whereas the maximum of the visible area is around 100%, i.e. the reference template occupied the whole camera image. The reference template is always facing the camera near fronto-parallel in these sequences, there is only rotation around the normal of the target.

The next type of sequences is “Fast Far”; here we go away from the target until it covers again an area of approximately 4% of the image. Then we move the camera with increasing speed resulting in big inter-frame motion. Towards the end of these sequences, the effect of motion blur shows strongly. These motions are also applied to the “Fast Close” sequences, the only difference here being that the reference image typically covers 60% and more of the image where parts of the targets go often outside the image.

The last type of sequences we recorded, “Illumination”, varies the lighting conditions of the scene while the camera is moving slowly. For this, we switch off and on again two sets of fluorescent tubes during the sequence, additionally a shadow is cast by a waving hand onto the reference target. In this scenario, the target is always covering more than 15% of the camera image.

Selected frames from the sequences can be seen in figure 3. Every sequence consists of 1200 RGB images with a resolution of 640×480 pixels acquired from the camera at 40 Hz, the measurement arm provides its absolute poses with 75 Hz. These data were recorded directly into the main memory of the attached computer in order to minimize the influence of slowing down the recording at arbitrary moments because of hard disk access. The images and poses are written to disk as batch job after each sequence. We also chose not to fuse the images with the poses while recording, instead we synchronize them offline. Thus, we make sure that also fast and sudden motions are accurately represented in the dataset.

5 Post-processing the dataset acquisition

After recording the images and poses of all 40 sequences, we still had to assign a specific pose to each image and compute the residual error of the sequences. Figure 4 shows an overview of the transformations in our setup which will be discussed briefly.

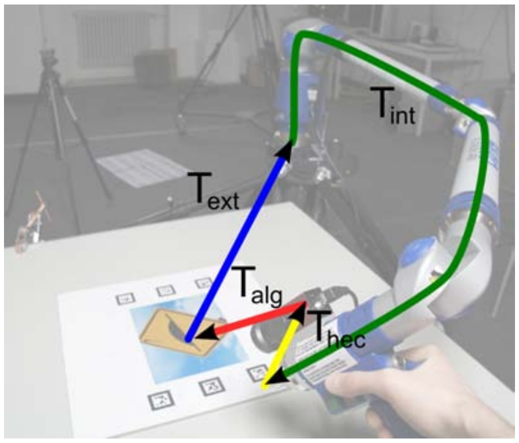
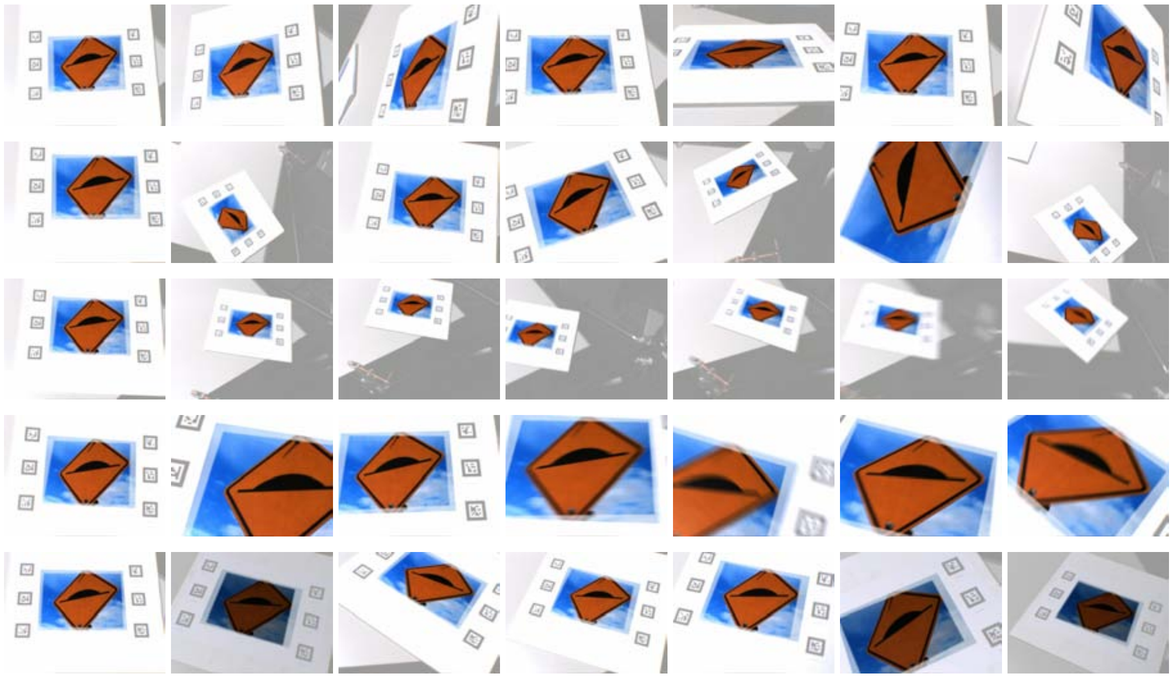


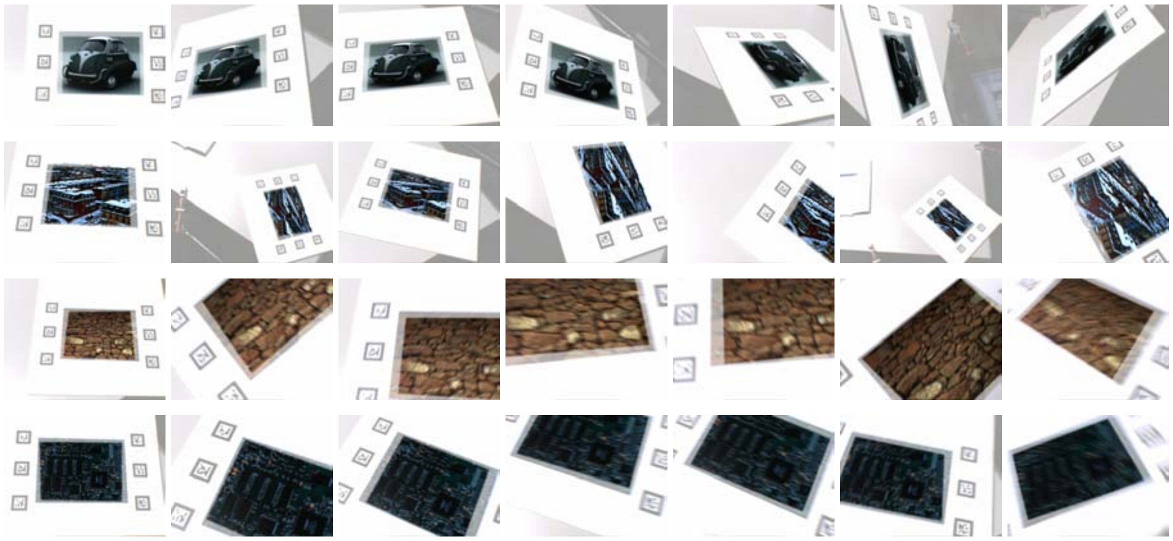
Figure 4: The transformations used in our setup.

The internal pose $\mathbf{T}_{int}$ of the robot is known to be very precise. The hand-eye-calibration $\mathbf{T}_{hec}$ is based on the AICON calibration pattern and the internal pose of the robot, thus it is also very precise. The transformation $\mathbf{T}_{alg}$ is given by the evaluated algorithm and compared to the ground truth pose computed via a concatenation of $\mathbf{T}_{hec}$, $\mathbf{T}_{int}$ and $\mathbf{T}_{ext}$. The external offset of the measurement arm $\mathbf{T}_{ext}$ has to be provided by generating 3D-3D correspondences with the tip of the arm by moving it to coordinates of known 3D reference points. We chose the 24 corners of six fiducials next to the reference target for those points. These corners were also used for computing the residuals, for the synchronization of the measurement arm with the camera and also for fine-tuning $\mathbf{T}_{ext}$. Although the detection of the fiducials is not perfect, it is extremely accurate [14] and thus appropriate for these tasks.

To synchronize the measurement arm to the images of the camera, we did the following: As soon as an image is fully captured from the camera, i.e. available to our software, we attach a timestamp t^i_{img} to it; the same is done with the poses of the measurement arm, these timestamps are denoted by t^j_{meas}. We assume that there is a constant offset t_{offset} between the timestamps t^i_{img} and t^j_{meas} that can be thought of the time needed to capture the image, to transfer the image information via the various busses to the main memory and finally to invoke all hard- and software interrupts until the image is available to our software. The optimal offset together with the optimal $\mathbf{T}_{ext}$ should give the lowest residual, i.e. in our case the lowest RMS of the reprojection error of the fiducals for a full sequence (1200 images). Due to the difference in the acquisition frequency between the camera and the measurement arm, the pose assigned to an image was interpolated from the two nearest neighbors using linear interpolation for the translational part and spherical interpolation for the rotational part. We used the Nelder-Mead algorithm [15] to jointly optimize t_{offset} and $\mathbf{T}_{ext}$ for each sequence, starting with $t_{offset} = 0$ and with the result of the manual registration for $\mathbf{T}_{ext}$.



(a) Target “Bump Sign”, from top to bottom: Sequence “Angle”, “Range”, “Fast Far”,“Fast Close” and “Lighting”.



(b) Targets “Isetta”, “Philadelphia”, “Wall” and “Lucent”

Figure 3: Sample images taken from some of the sequences each 200 frames. In this figure, the real background is blended with the image the algorithms process during the evaluation to highlight the randomized borders used.

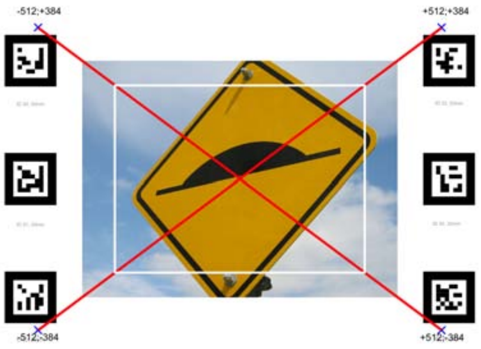


Figure 5: A reference target with the six fiducials. The inner area of each template is provided to the algorithms as reference image, they have to compute the position of the four points on the diagonals.

After the synchronization was completed, we arrived at an average RMS reprojection error of 0.86 pixels for all sequences. This error also incorporates all errors from the internal and external calibrations. The “Fast Close” sequences typically have the highest residual error (mean 1.54 pixels) due to the motion blur and the large size of the reference target in the image, the lowest residual error (mean 0.54 pixels) was found for the “Illumination” sequences.

6 Evaluating template-based tracking algorithms

We differentiate between tracking-by-detection and frame-by-frame tracking algorithms: Frame-by-frame tracking algorithms require an initial pose/homography, but then should be able to track the reference target in the images with high precision and over many consecutive frames. Similar to the Middlebury datasets [4], we are not planning to publish the ground truth for every frame. However, to support the frame-by-frame algorithms, we give the pose/homography for the first and every 250th frame in the dataset. Thus it is possible to initialize a frame-by-frame algorithm five times per sequence in case it lost tracking.

However, tracking-by-detection algorithms were provided with the reference images only, they had to locate the target in the images of the sequences without any further prior knowledge about its pose. The missing need of an approximate initial pose is a benefit over the frame-by-frame tracking algorithms, they are usually initialized using detection-style algorithms. The total number of sucessfully detected frames is more important for this type of algorithms than the number of consecutively tracked frames.

We evaluated four popular tracking algorithms: SIFT [10], SURF [5], FERNS [13] and ESM [6]. These will be discussed in the next sections.

6.1 Evaluation

The evaluation of the four algorithms was conducted as follows. Each algorithm was given the 640×480 uncompressed image of the reference targets, more specifically of the centered inner area of each reference targets (marked white in figure 5). Then all algorithms were given time to transform the data in the format suitable for tracking, i.e. for SIFT and SURF the descriptors of the features of the reference images were constructed, for FERNS the classifiers were trained.

After that they were given the undistorted images of the sequences. To prevent algorithms from being distracted by a cluttered background, we replaced the background with white. In addition to that, we also removed the original borders of the reference target to prevent algorithms from simply using the image borders instead of the template image itself. The original borders of the 800×600 reference target were replaced by randomized borders, but at the same time we made sure that the 640×480 image the algorithms were given is not cut by the new randomized borders. Figure 6 visualizes these steps.

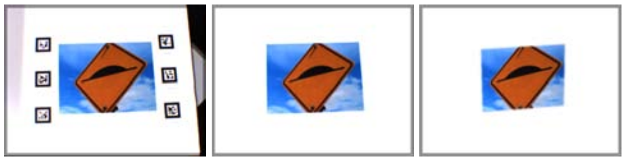


Figure 6: The steps taken to prepare an image for evaluation: Left the captured image, in the middle the undistorted image with the background removed, on the right the final image with randomized borders.

While removing the background is clearly a simplification, we chose this approach with the current datasets to focus on the best performance of the algorithms possible with images of the tracked object coming from a real camera. In later datasets we could add a cluttered background to the real scene or add a virtual cluttered background after recording.

The evaluation is based on four reference points which are placed on the diagonal lines of the reference images (marked with blue crosses in figure 5); they are at the XGA resolution boundaries, i.e. at $(\pm512;\pm384)$. For every image $\mathbf{I}_i$ per sequence, the RMS distance err_i of each imaged reference point $\mathbf{x}_j$ to the ground truth point $\mathbf{x}_j^\star$ was computed as

$$err_i = \sqrt{\frac{1}{4}\sum_{j=1}^{4}\|\mathbf{x}_j - \mathbf{x}_j^\star\|^2}$$

After computing these errors for a sequence, all frames with an $err_i \geq 10$px are removed as we regard the cases with a higher RMS error as sign that the tracking algorithm lost the target. Based on these filtered results, we compute the ratio of tracked frames and analyze the distribution of the error.

6.2 Results

We used the original implementations of ESM, FERNS and SURF for the evaluation, for SIFT we used the implementation from Vedaldi and Fulkerson [19]. The majority of the parameters were left at their authors’ default settings, we only constrained SURF and FERNS to use the 800 strongest points, a number we had to provide to the implementations that was high enough to not degrade their performance. To compute the poses with the feature-based algorithms, we used nearest-neighbour matching to generate 2D-3D correspondences, then removed outliers via RANSAC and finally computed a pose via DLT that was refined with Levenberg-Marquardt. For ESM, we used the output homography to project the corners of the reference template into the current frame and also computed a pose via DLT and Levenberg-Marquardt.

The targets were evaluated in the order as shown in figure 2, i.e. “Low”, “Repetitive”, “Normal”, “High Texturedness”. Each target was evaluated following the discussed foci in the order “Angle”, “Range”, “Fast Far”, “Fast Close”, “Illumination”. The results of the evaluation indicated that SIFT is, most of the time, outperforming FERNS and SURF in terms of accuracy and percentage of tracked frames. However, it should be mentioned that the evaluation of SIFT took more than 2.5 days (approximately 3 s per frame) to compute whereas FERNS, SURF and ESM finished in less than 6 hours each.

The focus of the evaluation in this paper is primarily to see whether the targets and the chosen sequences per target were suitable for building a dataset that is both challenging and at the same time not undoable so that it can be useful to the computer vision community. That is why we concentrated on the accuracy of the algorithms in close-to-ideal situations. The timings were not considered since the best possible framerates of the algorithms are generally achieved with extensive finetuning of parameters and this was of minor interest to us.

In figure 7 the percentage of correctly tracked frames is given for all algorithms and targets. The evaluation showed that ESM often depends on the selected area to be tracked. In contrast to the feature points approaches that typically select positions with high cornerness, ESM gives the same weight to every pixel in its area. This can severely degrade the accuracy if e.g. the border of the image is approximately uniform. Thus, to make the comparison fair, we manually selected patches in the low texture targets for ESM. After that, it tracked the extremely low textured yellow road sign for 100% of the "Angle" sequence, also surpassing all three other algorithms in terms of accuracy. Concerning low texturedness, both FERNS and SURF showed a better performance on the slightly more textured stop sign target. The reason for the performance of SURF for the first target is that SURF does not find sufficient feature points on the yellow traffic sign, the same again applies to the grass target which for ESM also turned out to be an extremely difficult target. FERNS was in general very well adapted to the "Angle" sequences which might come from the explicit training phase that warps the reference targets numerous times.

The "Fast Close" sequences with large amounts of motion blur were the most difficult to detect for all four algorithms, whereas "Range" and "Illumination" sequences were often correctly detected. Figure 8 shows the RMS errors for all targets, sequences and algorithms as box-and-whiskers-like diagram; the whiskers mark the minimum and maximum error while the box spans from the first to the third quartile, the mean is given via the red horizontal line. The targets per target group are separated by a vertical black line. In general, the "Fast Close" sequences were detected with the largest error per target while "Illumination" yielded, most of the time, the lowest error.

7 Conclusion

We presented a methodology to create a dataset for evaluating template-based tracking algorithms. The goal was to create image sequences with precisely known poses of the camera so that they can be used as objective ground truth to evaluate algorithms and enable fair comparisons.

The ground truth sequences were recorded using a highly precise measurement arm together with an industrial camera. They feature realistic imaging conditions and motions and are very precise. When generating the dataset, we carefully selected the texture of the chosen targets and the camera motions to be as much representative as possible. Using these sequences, we evaluated four state-of-the-art algorithms. Our sequences can now be used by the vision and AR communities, we offer a webpage at *http://metaio.com/research* to give other authors the opportunity to extensively evaluate their template-based tracking algorithms and to be able to objectively compare them to other methods. By using not every consecutive image of the sequences, but only every *n*th image, it is also possible to analyze the effect bigger interframe motions, or even for wide-baseline matching.

The dataset is not frozen and it is meant to be evolutive, e.g. we would be able to add new sequence types in case it is requested by a large number of users. Although we focussed on planar reference targets, extending the presented approach in order to evaluate 3D tracking methods is straight forward.

Acknowledgements

This work was partially supported by BMBF grant Avilus / 01 IM08001 P.

References

[1] AICON 3D Systems GmbH. http://aicon.de/.

[2] Allied Vision Technologies GmbH. http://alliedvisiontec.com/.

[3] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56(3):221–255, 2004.

[4] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *ICCV*, pages 1–8, 2007.

[5] H. Bay, T. Tuytelaars, and L. J. V. Gool. Surf: Speeded up robust features. In *ECCV*, pages 404–417, 2006.

[6] S. Benhimane and E. Malis. Homography-based 2d visual tracking and servoing. *International Journal of Robotic Research*, 26(7):661–676, 2007.

[7] FARO Europe GmbH & Co. KG. http://faro.com/.

[8] D. J. Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America A*, 4(8):1455–1471, 1987.

[9] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *ISMAR*, pages 225–234, 2007.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[11] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[12] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3):263–284, 2007.

[13] M. Özuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *CVPR*, pages 1–8, 2007.

[14] K. Pentenrieder, P. Meier, and G. Klinker. Analysis of tracking accuracy for single-camera square-marker-based tracking. In *Proc. Dritter Workshop Virtuelle und Erweiterte Realität der GI-Fachgruppe VR/AR*, 2006.

[15] C. J. Price, I. D. Coope, and D. Byatt. A convergent variant of the nelder-mead algorithm. *J. Optim. Theory Appl.*, 113(1):5–19, 2002.

[16] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, pages 519–528, 2006.

[17] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *PAMI*, 30(6):1068–1080, 2008.

[18] R. Y. Tsai and R. K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5(3):345–358, 1989.

[19] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[20] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *ISMAR*, pages 125–134, 2008.

[21] K. Zimmerman, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *PAMI*, 31(4):677–692, April 2009.

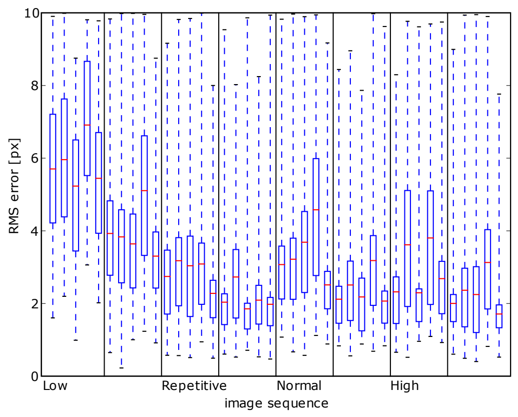
FERNS	Angle	Range	Fast Far	Fast Close	Illumination
Low	17,08%	8,08%	1,58%	3,75%	6,58%
	47,33%	71,00%	22,92%	40,50%	76,08%
Repetitive	36,42%	65,17%	15,42%	48,50%	91,83%
	42,50%	45,17%	6,25%	50,00%	81,33%
Normal	69,50%	80,58%	24,92%	68,00%	95,92%
	38,75%	53,08%	9,00%	64,67%	81,67%
High	34,92%	38,17%	5,92%	16,00%	31,58%
	71,75%	61,50%	13,42%	63,00%	96,92%

SIFT	Angle	Range	Fast Far	Fast Close	Illumination
Low	47,25%	49,08%	10,33%	19,58%	59,17%
	36,08%	95,42%	25,50%	55,58%	99,75%
Repetitive	59,00%	99,33%	43,33%	71,92%	100,00%
	69,50%	95,67%	15,17%	62,83%	98,17%
Normal	63,50%	84,25%	21,75%	55,17%	96,08%
	53,00%	96,08%	31,67%	77,67%	99,58%
High	66,83%	85,08%	18,33%	37,42%	97,00%
	79,50%	94,75%	31,42%	72,75%	99,50%

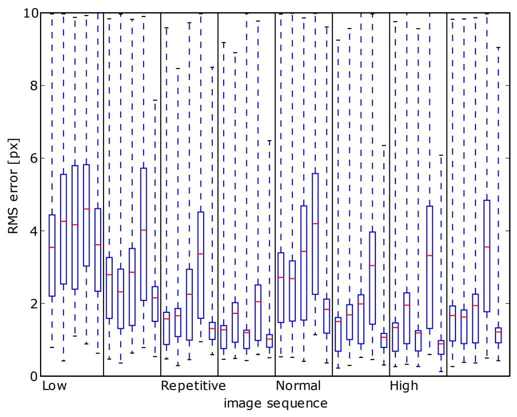
SURF	Angle	Range	Fast Far	Fast Close	Illumination
Low	0,50%	0,33%	0,08%	0,00%	0,00%
	27,17%	67,00%	11,83%	33,50%	55,17%
Repetitive	16,50%	38,42%	5,25%	47,33%	41,00%
	25,58%	50,00%	6,08%	54,17%	49,50%
Normal	37,92%	50,17%	6,17%	50,33%	67,75%
	45,33%	70,75%	14,25%	69,67%	89,58%
High	0,00%	7,75%	0,00%	0,08%	0,00%
	64,00%	44,42%	6,50%	51,50%	72,33%

ESM	Angle	Range	Fast Far	Fast Close	Illumination
Low	100,00%	92,33%	35,00%	21,58%	71,08%
	100,00%	64,17%	10,58%	26,83%	56,25%
Repetitive	61,92%	50,42%	22,50%	50,17%	34,50%
	2,92%	11,33%	6,83%	35,83%	11,33%
Normal	95,42%	77,75%	7,50%	67,08%	76,75%
	99,58%	99,00%	15,67%	86,75%	90,67%
High	0,00%	0,00%	0,00%	0,00%	0,00%
	100,00%	61,42%	22,83%	45,50%	79,67%

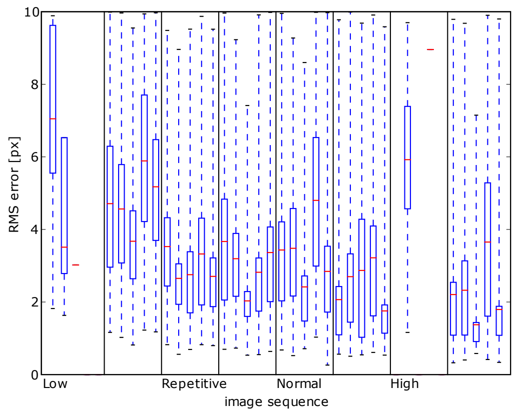
Figure 7: Ratio of successfully tracked images (with $err_i < 10$ pixels).



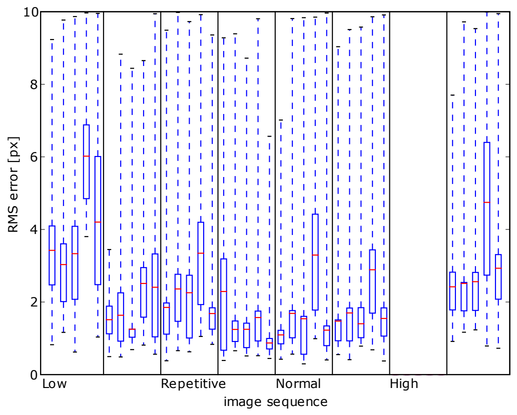
(a) FERNS



(b) SIFT



(c) SURF



(d) ESM

Figure 8: Distribution of the RMS error for each sequence, only successfully tracked frames were taken into account. The whiskers denote minimum and maximum, the box spans from first to third quartile; a red line segment shows the mean RMS error.