Learning to Track with Multiple Observers

Björn Stenger
Computer Vision Group
Toshiba Research Europe

Thomas Woodley
Dept. of Engineering
University of Cambridge

Roberto Cipolla
Dept. of Engineering
University of Cambridge

Abstract

We propose a novel approach to designing algorithms for object tracking based on fusing multiple observation models. As the space of possible observation models is too large for exhaustive on-line search, this work aims to select models that are suitable for a particular tracking task at hand. During an off-line training stage observation models from various off-the-shelf trackers are evaluated. From this data different methods of fusing the observers on-line are investigated, including parallel and cascaded evaluation. Experiments on test sequences show that this evaluation is useful for automatically designing and assessing algorithms for a particular tracking task. Results are shown for face tracking with a handheld camera and hand tracking for gesture interaction. We show that for these cases combining a small number of observers in a sequential cascade results in efficient algorithms that are both robust and precise.

1. Introduction

It is well known that combining multiple observation models can significantly improve the performance of a visual object tracker. The literature on multi-cue tracking essentially demonstrates the concept of different cues complementing each other and thus overcoming the failure cases of individual cues [2, 3, 8, 11, 15, 18, 20, 21, 23]. A typical example might be a hand being tracked while it moves in front of the face. The hand may still be tracked based on shape features while color features become less reliable. The most common approach to multi-cue tracking is to evaluate several observers in parallel and subsequently combine their output, by either switching between them [2] or by probabilistically merging them [8, 15, 18, 20]. The main issue when merging tracking results is how to obtain a good confidence measure for each cue. This is a tricky question since the performance of one cue may only be assessed by using a different cue or different representation of the target object. One answer is the discriminability between foreground object and background region. This is the basis of recent work on *discriminative* tracking [1, 4, 9], where tracking is formulated as a classification task. Collins *et al*. proposed a method for on-line feature selection which selects the most discriminative features from a pool of color-based features [5]. Discrimination was evaluated as either the two-class variance ratio or the difference of the first two likelihood peaks. Avidan introduced ensemble tracking, where multiple (3-5) weak classifiers are combined via AdaBoost [1]. At each frame a new weak classifier is learned and the ensemble is updated by replacing the least reliable classifiers in each time step. A variation on this theme is the on-line boosting tracking algorithm by Grabner and Bischof, where a larger pool of 250 weak classifiers is evaluated and updated at each time step and a smaller number of 50 selectors chooses the ones that are combined into a strong classifier [9].

In practice, an issue with on-line adaptation is the adaptability vs. drift trade-off. Allowing the tracker to adapt to rapid changes of the object's appearance brings the risk of incorrectly adapting to the background. Ideally one would like to have an object model available that includes all possible variations. Such a fixed object model could then be used as an 'anchor' for the tracker. Obtaining such a model is challenging and different representations have been used, including the color distribution [2], a representation learned from a short initial sequence [10] or an off-line trained detector [13, 25]. Detectors have been included into tracking systems by either simply running them in tandem [13, 25] or by integrating them within the tracker's observation model [16, 19]. Indeed, a viable tracking solution is to use a detect-and-connect strategy, shown for example in [12]. However in many cases this approach is not yet sufficiently fast for real-time tracking and the detectors are still not sufficiently flexible.

In this paper we address the question of how to design a tracker using multiple observation models. The idea is to learn which of these are suitable as components and how they should be arranged for efficient evaluation. We consider particular tracking scenarios, e.g. tracking a face with a handheld camera, and collect representative sequences that are ground-truth labeled by hand. We learn error distributions on the training set that are then used to efficiently

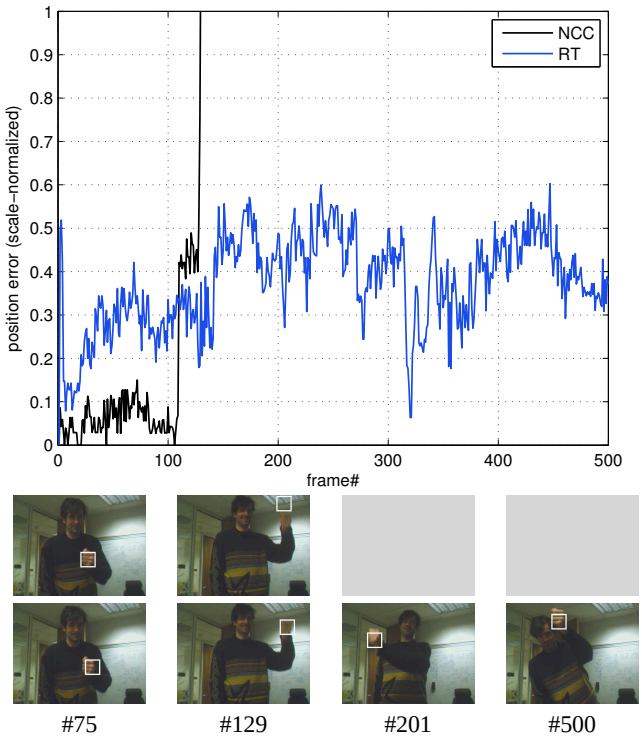

Figure 1. **Example of precision vs. robustness of trackers.** *The plot shows the tracking error on a hand tracking test sequence (below) of two stand-alone trackers with different observation models: maximum correlation (NCC), and randomized template tracking (RT). In this example NCC is more accurate but fails early on, while RT is able to track over a longer period with less precision.*

evaluate combinations of observers on the test set. The tracking algorithm therefore only needs to include a small number of components at run-time. The observation models are components from different stand-alone tracking algorithms such as single template matching, optical flow and on-line classification. We also include an off-line trained detection component that is used to initialize the tracker and prevent drift.

The following section introduces a method for evaluating individual observers, introducing notions of tracker precision and robustness. Section 3 explains how these measurements can be used for evaluating the performance of combinations of observers. Schemes for parallel as well as cascaded computation of the observers are compared. Experiments in section 4 show results on two scenarios, face tracking with a handheld camera and hand tracking with a static camera.

2. Evaluation of Observation Models

The goal is to find, for a given tracking scenario, the best observer or combination of observers. The approach is to first evaluate each observer individually and from these values measure the performance of combinations of observers. The observers we consider are those used previously in tracking algorithms, see Table 1 for a list of observers evaluated. They can be classified into four types: single template matching, motion consensus of local features [2, 13, 17], histogram-based region matching [6] and on-line classification [4, 9, 16]. Note that the individual observers are not restricted to using a single cue.

Given an image sequence $I_t, t = 1, ..., T$, at every time step t each observer $O^k, k = 1, ..., K$ computes an estimate of the target location $\hat{\mathbf{x}}_t^k$ as well as an error $e_t^k = d(\hat{\mathbf{x}}_t^k, \mathbf{x}_t^{gt})$ as distance to the labeled ground truth location $\mathbf{x}_t^{gt}$.

The estimate $\hat{\mathbf{x}}_t$ is represented by a center location and scale estimate and typical distance measures are either bounding box overlap or a scale-normalized distance between the centers [5]. Every observer also outputs a confidence value c^k, which is computed depending on the type of observer. Following previous work, this can be a histogram distance for region trackers [6], a measure of motion consensus for local feature trackers [2] or the classification margin for on-line classifiers [1]. Confidence values have regularly been used to compare and integrate the results of multiple observers. However, most observers have a relatively simple object representation thus the confidence value itself cannot be expected to be perfectly reliable. For example an observer may have a high confidence value at an incorrect location if there is an object close-by that is similar to the target in the observer's feature space. Here the confidence value is simply regarded as a single feature computed by the observer. Loss of track occurs when the error e_t^k is above a threshold value τ. In this case the tracker outputs τ as error value and is re-initialized at the next successful detection. Detections are pre-computed by running an off-line detector over all sequences. The performance of a tracking algorithm is estimated as the expected error over all frames

$$E[e^k] = \frac{1}{T} \sum_t e_t^k, \ k = 1, ..., K. \tag{1}$$

However, this function does not allow the comparison of observers when track is lost because the error is meaningless in this case. In practice we are therefore interested in both the tracking error while the tracker is following the target as well as the probability of losing track. This motivates the distinction into two performance criteria, precision and robustness. Precision is related to the expected error during successful tracking by

$$1 - E[e^k | e^k < \tau]. \tag{2}$$

The robustness is the probability of successful tracking as

$$p(e_t^k < \tau | e_{t-1}^k < \tau). \tag{3}$$

The measurements for the individual observers O_k on a training sequence are thus given by

$$\mathcal{Z}^k = \{\hat{\mathbf{x}}_t^k, e_t^k, c_t^k\}, \quad t = 1, ..., T\,. \tag{4}$$

This allows the evaluation of single observers on the complete sequence, not just on the first successfully tracked segment. Note that loss of track can occur at any time during the sequence when an observer's particular assumptions, e.g. slow motion or small pose change, do not hold. The number of tracked frames when running the tracker only once is dependent on when this event occurs: if it is near the beginning of a sequence the measured robustness is worse than when it is near the end. The advantages of the proposed measure are the following: (i) it is independent of the position in time of failure cases. (ii) all frames in the labeled sequences are considered in the evaluation. Frames during tracking failure are discarded, so the underlying assumption is that the time until the next detection is relatively short.

It is also interesting to consider the relationship between precision and robustness. Observers with a fixed spatial model tend to be more precise than observers where the spatial arrangement is more flexible, see for example Figure 1 which shows tracking (without re-initialization) using single template matching and local feature matching on one of the test sequences. Note that similar ideas have recently emerged in the visual object classification literature, where a representation's invariance vs. discriminative power trade-off was explored [24].

It is possible to use each observer as a single component in a tracking algorithm. Without using a stopping criterion, the tracker will continue tracking and eventually lose the target. For unseen data a threshold on the confidence value is commonly used to terminate tracking. If ground truth is available, the trade-off between precision and robustness can be explored by changing the threshold value τ (in equations 2 and 3). When the tracking error exceeds τ, tracking is stopped and the tracker re-initialized. Setting τ to a small value enforces high precision but low robustness and vice versa. In our experiments for evaluating individual observers the threshold value τ on the error is set to 1, corresponding to the case of having clearly lost track. Precision and robustness are then computed from the measurements (4) as in equations 2 and 3, taking expectations over all frames of the test sequences.

3. Evaluating Multiple Observers

This section deals with the question of how to evaluate the performance gain that can be achieved by combining multiple observers. We distinguish two different approaches of combining observers: parallel and sequential, respectively. In parallel evaluation the estimates of multiple observers are available at each time step and the output of the most reliable observer is selected. Alternatively, other fusion methods could also be used, for example by weighting the observer estimates or voting schemes that give a consensus-based estimate. In sequential or cascaded evaluation observers are evaluated in sequence: if the first observer returns a high confidence, then no other observer is evaluated. Otherwise the evaluation continues with the next observer. The advantage of sequential observation is that on average significantly less computation is required. However the order of evaluation as well as the thresholds on the confidence values clearly are critical for good performance.

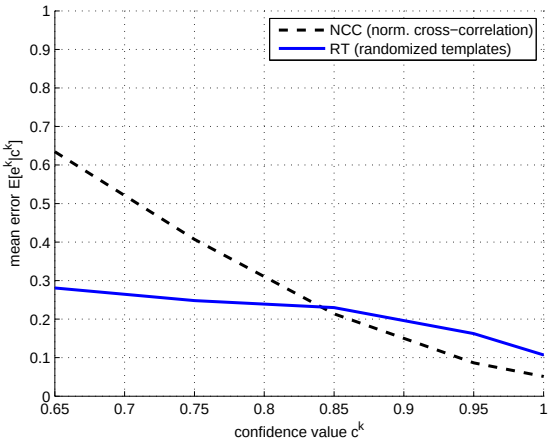

Figure 2. **Expected error as function of confidence.** *This data is obtained from training sequences and allows the direct comparison of observers given their confidence values. Shown here are the results for observers NCC (norm. cross-correlation) and RT (randomized templates).*

Given the individual measurements $\mathcal{Z}^k$ of observers we evaluate the method of switching observers based on their confidence c^k. Ideally we would like to select the observer with the lowest error at each time step. This information is not available at test stage, so instead the observer's confidence value is used. However, these values of different observers cannot be compared directly, since they are computed in different ways (see last column in Table 1). In order to make these values comparable we estimate the distributions $p(e^k|c^k)$ from the training data, i.e. the error distribution given the confidence value of observer O^k. To use the finite data set we discretize the range of the c^k values and compute $p(e^k|c^k)$ in each partition. For the evaluation we represent it by the mean of each distribution, $E[e^k|c^k]$. Figure 2 shows two of these functions for observers NCC (normalized cross-correlation) and RT (randomized templates) on hand tracking data. As an example, if both observers returned a confidence value of 0.9, the expected error of NCC is lower than that of RT and NCC should be chosen.

Method	Observation	Estimate	Confidence value
NCC	Normalized cross correlation	max correlation	correlation score
SAD	Sum of absolute differences	min distance	distance score
BOF	Block-based optical flow of 3 × 3 templates	mean motion	mean NCC score
KLT [17]	Kanade-Lucas-Tomasi sparse optical flow using 50 features	centroid of good features	fraction of good features
FF [13]	Flocks of features: Tracking 50 local features with high color probability and 'flocking' constraints	centroid of good features	fraction of good features
RT [2]	Randomized templates: NCC track of eight subwindows, with motion consensus and resampling	centroid of good features	fraction of good features
MS [6]	Mean shift: Color histogram-based mean shift tracking with background weighting	min histogram distance	histogram distance
C [22]	Color probability map, blob detection	scale space maximum	probability score
M [22]	Motion probability map, blob detection	scale space maximum	probability score
CM [14]	Color and motion probability map	scale space maximum	probability score
OBD [9]	On-line boosted detector: Classifier boosted from pool of rectangle features updated on-line	max classifier output	classifier margin
LDA [16]	LDA classifier computed from five rectangle features in the previous frame (Observer 1 in [16])	max classifier output	classifier margin
BLDA [16]	Boosted LDA classifier using 50 LDA classifiers from a pool of 150, trained on the previous five frames (Obs. 2 in [16])	max classifier output	classifier margin
OFS [4]	On-line feature selection of 3 out of 49 color-based features based on fg/bg variance ratio	centroid of top features	mean variance ratio of selected features

Table 1. **Observers in the evaluation.** *A diverse range of observers are tested in the experiments. They can roughly be grouped into four types: single template matching, local feature matching, histogram-based region matching, and on-line classifiers. Between them they use a variety of cues, including image intensity, color and motion features. Some observers maintain a fixed representation while others are updated over time.*

3.1. Parallel evaluation

The parallel evaluation scheme selects the observer with the lowest expected error given its confidence value at each time step, i.e. $k^* = \text{argmin}_k \; E[e^k|c^k]$, see top of Figure 3. If this error is above a certain threshold, then a detector is used to re-initialize. The output of the individual observers is used to evaluate the performance over different combinations of observers.

The running of tests consisting of all possible combinations of all trackers on all test sequences would take a prohibitive amount of time to complete. We therefore run all the observers individually on the test sequences and record the results for each frame. These results are then used in the combination tests as the result from each component observer. In order to test the validity of such a setup, we performed tests using the complete tracking framework for selected combinations of observers.

3.2. Cascaded evaluation

Although the combined estimate is expected to be better than individual estimates, the main disadvantage is the increased execution time. In cascaded evaluation observers are evaluated in sequence, starting with the first observer, and continuing with the next observer only if the expected error is above a threshold value, see Figure 3 bottom. If no observer returns a sufficiently low expected error, the algorithm attempts to jump to the top of the cascade using local detection. For evaluation, the output of the individ-

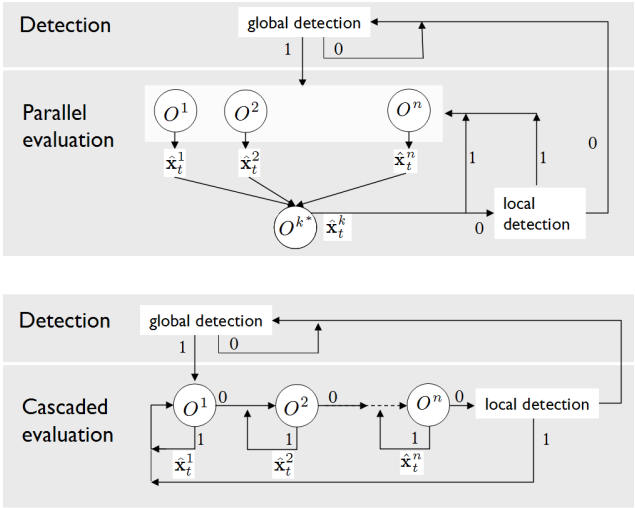

Figure 3. **Evaluation schemes.** *(top) In the parallel evaluation the output from the observer with the lowest expected error is chosen. (bottom) In the cascaded evaluation the next observer is only evaluated if the expected error is above a threshold. An off-line trained detector is used to re-initialize. The binary tests in this schematic represent threshold tests on the expected error.*

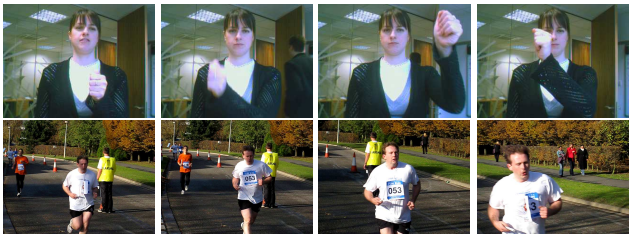

Figure 4. **Example sequences.** *The algorithms are tested on (top) hand tracking on sequences taken indoors with a static camera and (bottom) face tracking on footage taken outdoors with a handheld camera. Both datasets contain motion blur and pose changes of the target.*

ual observers is used to assess the performance of different combinations of observers and threshold values for switching observers.

3.3. Dynamic model discussion

A dynamic model is an integral component in every tracking algorithm as it can enable tracking through short periods of occlusion or weight the observations according to the most likely target motion. However, for our evaluation we do not want to be dependent on the dynamics which are difficult to model in the case of rapid hand motion or camera shake. Instead we sample the observation space densely at each pixel location in a neighborhood around the previous estimate and rely only on the observations without any prediction. This methodology is consistent with the observation made in the particle filtering literature that the performance largely depends on the proposal distribution [7]. No dynamic model is used in our experiments, corresponding to a maximum likelihood location estimate.

4. Experimental Results

We evaluated the method on two datasets, featuring hands and faces, respectively. The hand dataset contains 12 sequences (10 with rapid motion, 2 with slower motion) of 500 frames each of size 320×240, recorded at 30 fps [22]. The sequences are taken indoors with a static camera on top of a screen showing different people pointing their fist towards the camera in order to control a screen pointer. The face sequences each show a runner approaching the camera during an outdoor relay. The dataset contains 42 sequences of 100 frames each of size 640×480, recorded at 30 fps with a handheld camera. Example frames of these two datasets are shown in Figure 4. The hand dataset contains motion blur, hand pose changes, other skin-colored objects and occasionally people moving in the background. Tracking challenges for the face sequences include head pose and expression changes, camera shake, cast shadows and motion blur. Half of the sequences are used to learn the expected errors for each observer and the other half is used for performance evaluation.

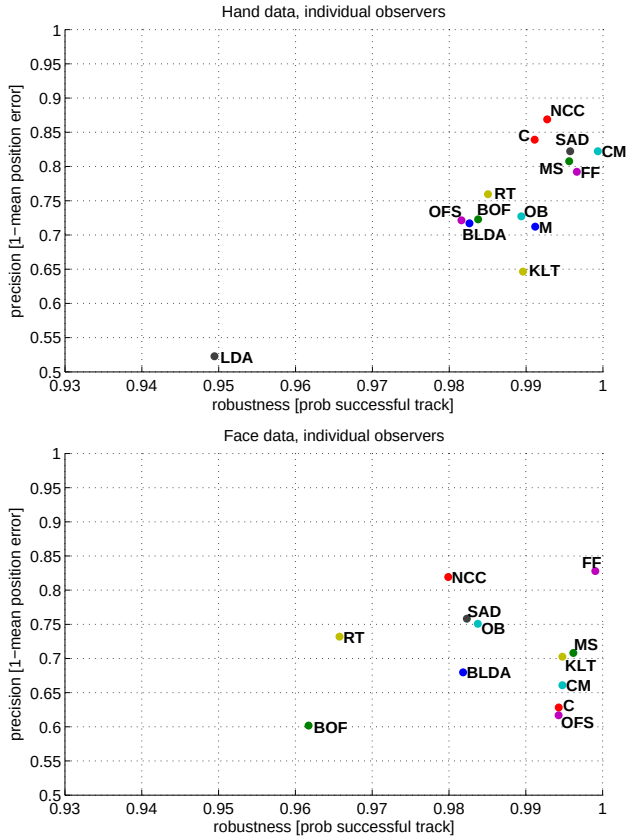

Figure 5. **Evaluating individual observers.** *These plots show precision and robustness on the test data for (top) hand data and (bottom) face data. NCC is the most precise observer on the hand dataset and the flocks-of-feature (FF) the most precise on the face data. In terms of robustness, the color-motion observer (CM) is best on the hand data while the FF observer performs best on the face data.*

4.1. Individual observers

We evaluated the observers in Table 1 on the two datasets. Figure 5 shows precision and robustness measurements on the unseen test sequences. For the hand data, NCC shows the highest precision while the color-motion (CM) observer is the most robust. On the face data the flocks-of-features (FF) observer, which models the target using a number of local features together with a color probability, is the most precise and robust observer. Single template observers such as NCC and SAD show lower robustness on the face data than on the hand data due to larger pose changes. Among the on-line classifiers on-line boosting (OB) shows the highest precision on both datasets. The LDA-based classifiers show relatively low precision on both datasets. The robustness values of LDA and motion (M) observers on the

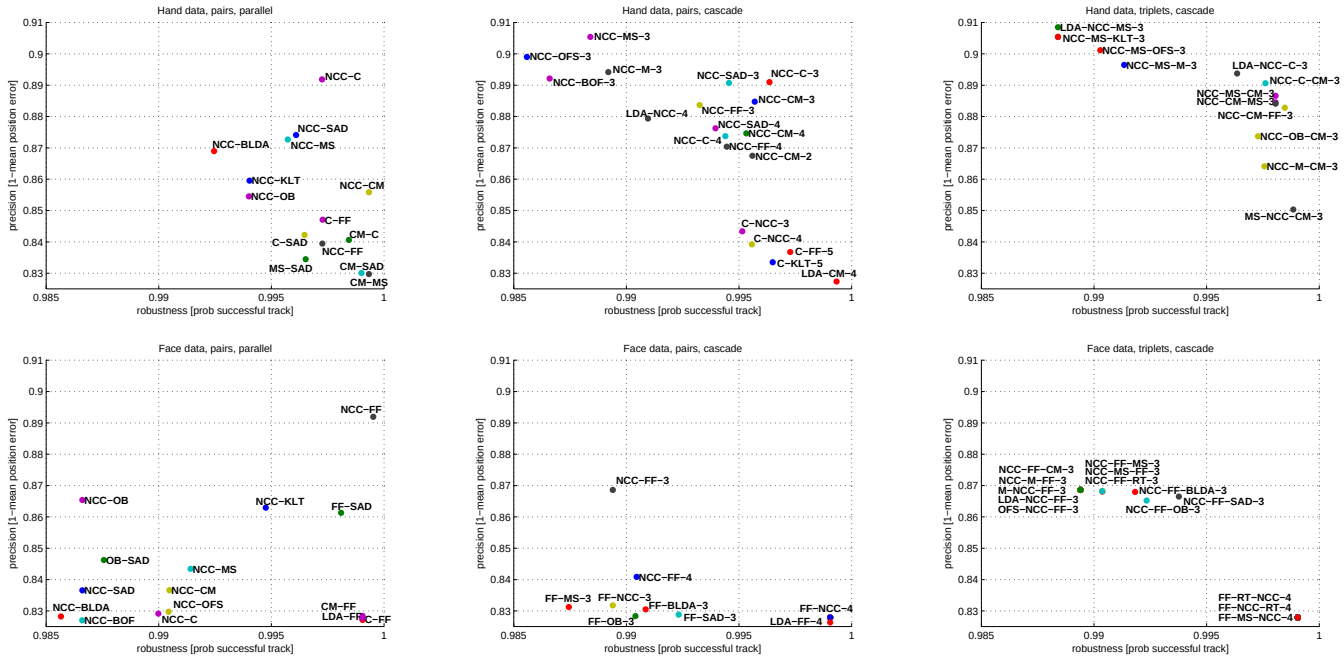

Figure 6. **Evaluation of observer combinations.** *These plots show the precision and robustness measured on the test sequences: (top row) hand data, (bottom row) face data. (left) pairs, parallel evaluation, (middle) pairs, cascaded evaluation, (right) triplets, cascaded evaluation. Only a small subset of data points near the upper right frontier with both high robustness and precision are shown here.*

face data are both below 0.93 and are not shown in the plot.

4.2. Parallel evaluation

We evaluated all pairs of observers using a threshold value of $\tau = 1$, giving a total of 91 combinations. Subsets of the results are shown in the left two plots of Figure 6. Only combinations are plotted that are near the upper right frontier of high robustness and high precision. On the hand data the combination of NCC with one of the color-based observers CM, C and MS shows good performance. In the videos the hand occasionally moves rapidly, resulting in significant motion blur. These cases tend to be failure modes for intensity or gradient based methods. On the other hand, the color distribution is less affected by motion blur. The robustness of these color-based observers is increased by most of the other observers that can help to bridge the frames where the color cue is unreliable. On the face data combinations of NCC with the local feature based observer FF is the most precise, while combinations of FF with many other observers are most robust. The analysis also shows how observers using different cues complement each other. For example on the hand data, the NCC-C combination has robustness-precision values of (0.997, 0.892), better than either NCC (0.992, 0.869) or C (0.991, 0.839) alone. Another example, which is not shown in the plot, is the combination of color (C) and local features (RT) on the face data, the same combination that was proposed in [2]. The combined observer C-RT has higher precision and robustness than either of the components alone.

4.3. Cascaded evaluation

We compared all ordered combinations of pairs at five different threshold levels (0.1, 0.2, 0.3, 0.4, 1.0) resulting in a total of 912 evaluations. Subsets of the results are shown in the two plots in Figure 6, middle. On the hand data most of the results with the highest precision employ NCC at the beginning of the cascade. High robustness is achieved when at least one of the observers uses the color cue, e.g. C or CM. The combination of NCC and CM that was proposed in [22] performs well in terms of precision, losing slightly in terms of robustness compared to the parallel evaluation. On the face data, the combination NCC-FF has the highest precision while FF-NCC is the most robust. The results also suggest that arranging the observers in the order of their individual precision leads to good performance. The idea is to estimate using the most precise observer at each time step. If the expected error falls below the threshold, the next observer essentially acts as a fallback method. Note that in some cases the cascaded tracker may have switched to an observer that is less precise during a difficult part of the sequence. It is therefore worth checking regularly if it is possible to jump to the top of the cascade again via local detection in order to increase tracking precision.

We also evaluated all triplets of observers at five different

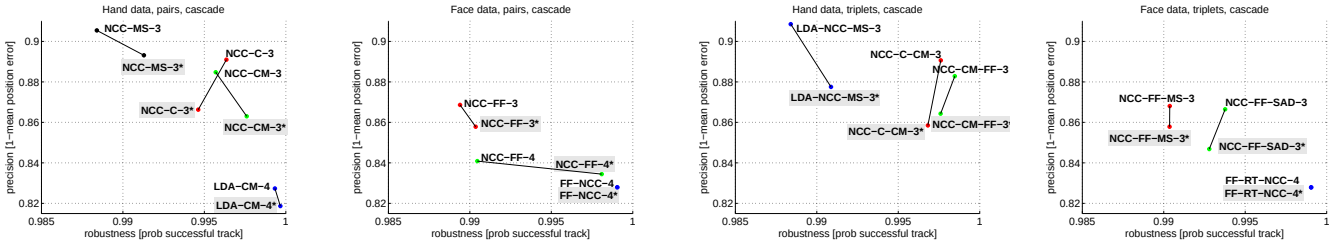

Figure 7. **Comparison with real tracking results.** *These plots show the precision and robustness measured for selected combinations of observers. It compares the results by theoretical combination (as in Fig. 6) and real tracking results obtained for selected combinations of observers, shown here with grey background. The left two plots show the results on pairs and the right two plots on triplets on the two datasets. The agreement is reasonable, although there is inherently some variation between the results. Note that these plots only show a small range of the complete data, thus the variation compared to the global values is small.*

threshold levels, a total of 4468 combinations. Subsets of the results are shown in the two plots in Figure 6, right. As a general observation, for both datasets the results are further improved. Successful combinations frequently include different types of observers, typically a single template, a color-based observer and either motion or local features. If one component is reliable over a long time period, the overall performance changes only little. This can be seen for example for the face dataset, where combinations of flocks-of-features (FF) as first component are consistently robust.

4.4. Tracker evaluation on selected combinations

Given that the above analysis of observer combinations is based on the analysis of individual observers, an obvious question is how this result varies when the full combination is tested in a tracking framework. They are not expected to give identical results because in this case the output of observers are dependent on each other. Testing all combinations of observers becomes prohibitively expensive, thus we use the results on independent observations as a method to select promising combinations to evaluate. Figure 7 shows results on pairs and triplets using cascaded evaluation on both datasets. The precision in the real tracking result is smaller than or equal to the results obtained with the simplified analysis. With few exceptions, e.g. the NCC-FF-4 cascade in the second plot of Fig. 7, the robustness values are very similar to the result.

Figures 8 and 9 show particular examples of observer switching on hand tracking. Figures 10 and 11 show examples on the face dataset.

5. Summary and Conclusion

This paper has presented a method for selecting suitable component observers for particular tracking tasks. To this end a comprehensive set of 14 observers has been evaluated on two challenging datasets. A new framework was proposed that evaluates the robustness and precision of observers, allowing the user to choose a profile suitable for a given application. The measurements of individual components were used to exhaustively evaluate combinations of components. We have shown results on observer pairs and triplets only, but the analysis can be applied to larger numbers of components.

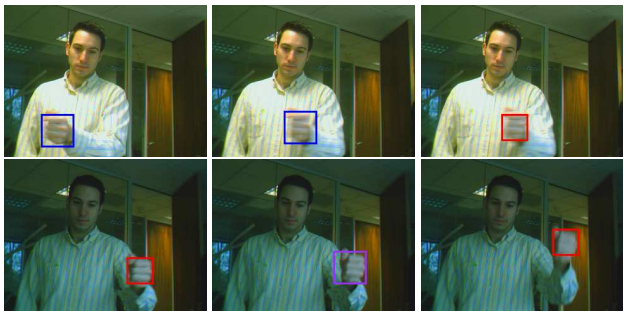

Figure 8. **Hand tracking using NCC-CM-M observers in parallel.** *The NCC-observer (blue) is used initially. During motion blur the tracker switches to the CM-observer (red). For a couple of frames the M-observer (purple) is used while the light is turned off before switching back to CM.*

The observers that were used in this paper have been used in stand-alone trackers. Some of these trackers themselves employ on-line feature selection. Here, instead of switching between relatively simple features from a finite pool, we propose switching on-line between observers that may use different cues and estimation schemes. Our evaluation framework allows combining arbitrary components that output an estimate and a confidence value. Direct comparison is possible because we estimate the observers' error distribution given their confidence.

In our experiments cascaded evaluation gives similar performance to parallel evaluation at much higher efficiency. One suggested strategy is to use the most precise tracker if possible and use more robust ones as a fallback mechanism, with an off-line trained detector for re-initialization. This architecture allows for long term operation, which is required in many applications.

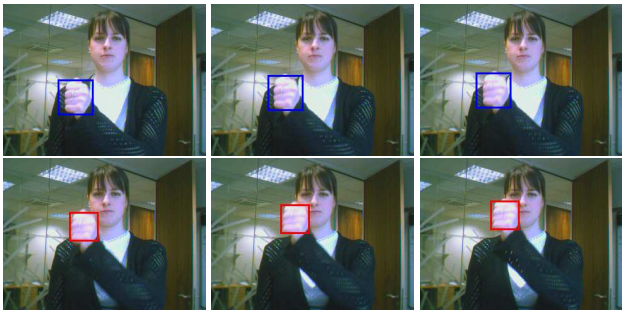

Figure 9. **Hand tracking using NCC-CM-FF observers in a cascade.** *The NCC-observer (blue) is used initially, switching to the CM-observer (red) during motion blur.*

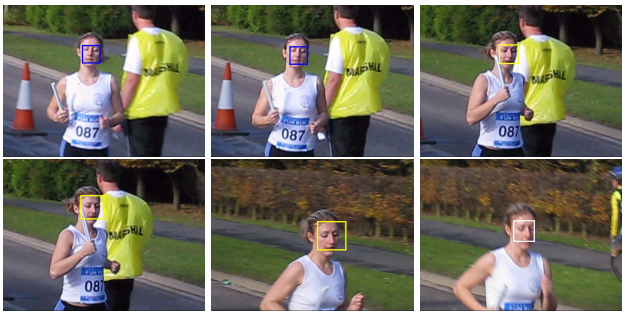

Figure 10. **Face tracking using an NCC-FF-MS cascade.** *Initially the accurate NCC-observer (blue) is used, switching to the more flexible FF-observer (yellow) as NCC can no longer handle the pose change. Note that there is local occlusion by the baton. In the end the included background area causes problems for FF and the tracker switches to color-based mean-shift (white).*

Figure 11. **Face tracking using an NCC-FF-MS cascade.** *NCC (blue) is used initially, switching to FF (yellow) when a strong shadow is cast on the face. Subsequently the tracker switches to mean-shift (white).*

References

[1] S. Avidan. Ensemble tracking. *IEEE Trans. Pattern Analysis and Machine Intell.*, 29(2):261–271, 2007. 1, 2

[2] V. Badrinarayanan, P. Pérez, F. Le Clerc, and L. Oisel. Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues. In *Proc. ICCV*, Rio de Janeiro, Brazil, October 2007. 1, 2, 4, 6

[3] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *CVPR*, pages 232–237, Santa Barbara, CA, June 1998. 1

[4] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *PAMI*, 27(10):1631–1643, October 2005. 1, 2, 4

[5] R. T. Collins, X. Zhou, and S. K. Teh. An open source tracking testbed and evaluation web site. In *Intl. Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, January 2005. 1, 2

[6] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *PAMI*, 25(5):564–575, 2003. 2, 4

[7] A. Doucet, N. G. de Freitas, and N. J. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001. 5

[8] W. Du and J. Piater. A probabilistic approach to integrating multiple cues in visual tracking. In *ECCV*, pages 225–238, Marseille, France, 10 2008. 1

[9] H. Grabner and H. Bischof. On-line boosting and vision. In *Proc. CVPR*, volume 1, pages 260–267, 2006. 1, 2, 4

[10] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *Proc. ECCV*, Marseille, France, October 2008. 1

[11] H. P. Graf, E. Cosatto, D. Gibbon, and M. Kocheisen. Multi-modal system for locating heads and faces. In *Proc. of the Second Intl. Conference on Automatic Face and Gesture Recognition*, pages 88–93, 1996. 1

[12] R. Kaucic, A. G. A. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs. A unified framework for tracking through occlusions and sensor gaps. In *CVPR*, pages 990–997, San Diego, June 2005. 1

[13] M. Kölsch and M. Turk. Fast 2D hand tracking with flocks of features and multi-cue integration. In *Workshop on Real-Time Vision for HCI*, Washington DC, July 2004. 1, 2, 4

[14] N. Krahnstoever, E. Schapira, S. Kettebekov, and R. Sharma. Multimodal human-computer interaction for crisis management systems. In *Proc. WACV*, pages 203–207, Orlando, FL, December 2002. 4

[15] I. Leichter, M. Lindenbaum, and E. Rivlin. A generalized framework for combining visual trackers – the black boxes approach. *IJCV*, 67(2):91–110, 2006. 1

[16] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. In *CVPR*, Minneapolis, MN, June 2007. 1, 2, 4

[17] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981. 2, 4

[18] F. Moreno-Noguer, A. Sanfeliu, and D. Samaras. Dependent multiple cue integration for robust tracking. *PAMI*, 30(4):670–685, 2008. 1

[19] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, volume I, pages 28–39, Prague, Czech Republic, May 2004. 1

[20] P. Pérez, J. Vermaak, and A. Blake. Data fusion for visual tracking with particles. *Proceedings of the IEEE*, 92(3):495–513, March 2004. 1

[21] M. Spengler and B. Schiele. Towards robust multi-cue integration for visual tracking. In *Machine Vision and Applications*, volume 14, pages 50–58, 2003. 1

[22] B. Stenger, T. Woodley, T.-K. Kim, C. Hernandez, and R. Cipolla. AIDIA - adaptive interface for display interaction. In *Proc. BMVC*, October 2008. 4, 5, 6

[23] K. Toyama and E. Horvitz. Bayesian modality fusion of multiple vision algorithms for head tracking. In *Fourth Asian Conference on Computer Vision*, Taipei, Taiwan, January 2000. 1

[24] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proc. ICCV*, Rio de Janeiro, Brazil, October 2007. 3

[25] O. Williams, A. Blake, and R. Cipolla. Sparse Bayesian learning for efficient visual tracking. *PAMI*, 27:1292–1304, 2005. 1