Hidden Hands: Tracking Hands with an Occlusion Aware Tracker

Akshay Rangesh, Eshed Ohn-Bar, and Mohan M. Trivedi
Computer Vision and Robotics Research Lab
University of California, San Diego
La Jolla, CA 92093-0434
{arangesh,eohnbar,mtrivedi}@ucsd.edu

Abstract

This work presents an occlusion aware hand tracker to reliably track both hands of a person using a monocular RGB camera. To demonstrate its robustness, we evaluate the tracker on a challenging, occlusion-ridden naturalistic driving dataset, where hand motions of a driver are to be captured reliably. The proposed framework additionally encodes and learns tracklets corresponding to complex (yet frequently occurring) hand interactions offline, and makes an informed choice during data association. This provides positional information of the left and right hands with no intrusion (through complete or partial occlusions) over long, unconstrained video sequences in an online manner. The tracks thus obtained may find use in domains such as human activity analysis, gesture recognition, and higher-level semantic categorization.

1. Introduction

Hand motions and gestures are an important cue for understanding and inferring human activity. In addition to this, they are an efficient form of high-bandwidth communication that may be used in conjunction with the verbal medium to convey important information. However, tracking hands and their articulations is not a trivial task. It is made complex by the large assortment of poses, shapes and sizes that hands take on in video sequences. These problems are further exacerbated by the rapid and often unpredictable movements of hands. Most approaches manage these concerns by constraining the experimental setup to ensure suitable operation for a given hand based application. These restrictions may be enforced by fixing the pose and location of the hands with respect to the camera, by conducting all experiments in a controlled environment (e.g. indoors), or by using multiple cameras to help with occlusions. The validity of these restrictions in a given scenario depends on the target application chosen.

Tracking of hand poses and hand-object interactions have been studied extensively in recent literature. These represent very rich streams of information in contexts of human computer interaction and human activity recognition. However, for certain applications, the motion of each hand as a whole in the image plane may be far more informative than other traditional hand-based information streams. This study is devoted to tracking both hands of a subject in the image plane, for applications in which hand motion and location may represent a useful cue for high level semantic applications. We use tracking both hands of a driver in an intelligent vehicle test-bed as an exemplar application for testing and evaluation.

To further motivate this study, we list a few potential applications below. First, hand tracking allows the study of preparatory movements for maneuvers[5, 17]. Such information may be useful when providing alerts and support to the driver[7]. A second potential application is in monitoring distraction levels, as hand-vehicle and hand-object interactions (such as text messaging, handling navigation, etc.) can potentially increase visual, manual, and cognitive load[11]. Because driver distraction is a leading cause of car accidents[23], studying where the hands are and what they do in the vehicle has never been a more pressing matter. A third possible application lies in providing a framework for hand gesture recognition and other high level activities[18] requiring accurate localization of hands. Finally, long term analysis of hand motion can provide useful insight into crash and near-crash events. For instance, in studying gestures performed by the driver for re-gaining control following an unexpected event. In addition to these domain specific applications, the tracking framework developed can be used to track two strongly interacting hands, riddled with frequent self occlusions, as is the case in ego-centric videos.

The main contributions of this paper are as follows: We propose a combined tracking-detection framework that operates online to provide short yet reliable tracklets, while data association is carried out using a bipartite matching algorithm. To handle frequent self occlusions between strongly interacting hands, we accommodate a mo-

Figure 1: Frames from the presented hand occlusion dataset. As can be seen, occlusions may occur at different positions for both the left and right hands.

tion matching algorithm that tracks hands through occlusion windows. We additionally introduce a challenging hand tracking dataset (complimentary to the VIVA challenge dataset[22]) with frequent occlusions, on which the proposed tracker is evaluated.

2. Related Work

To the best of our knowledge, there is no existing work that addresses the problem of tracking two or more interacting hands during naturalistic driving scenarios. We therefore provide a brief overview of works on single-hand tracking and generic multi-object tracking (MOT).

The 3-D model-based methods[21] for hand tracking can acquire in-depth and accurate motion data and are capable of coping with occlusions. However, these methods usually require a complex and expensive hardware setup, suffer from high computational cost and require dense representations of hand articulations. Blob-based approaches[1] detect hands as image blobs in each frame and temporally correspond blobs that occur in proximate locations across frames. Kalman filtering has been employed in works like[3] to transform observations (feature detection) into estimations (extracted trajectory). The advantages are real-time performance, treatment of uncertainty, and the provision of predictions for the successive frames.

Multi-object trackers, on the other hand, approach the problem from a data association standpoint. For instance, it is common for these algorithms to stitch small tracklets to produce "smooth" global tracks. However, this assumption is invalid in our situation where hand dynamics are highly erratic and almost never smooth e.g. a drivers' hand alternating rapidly between the wheel and gear stick. Moreover, most of these algorithms are modeled as data association problems that require tracklets generated beforehand. These are called *batch* or offline methods. It is difficult to apply such batch methods to time-critical applications such as ours where safety is of the essence. On the other hand, *sequential* or online methods like[24, 10, 8] attempt to resolve ambiguities in each frame (or in a small time window). However, considering more frames before making association decisions should generally help better overcome ambiguities caused by longer-term occlusions and false or missed detections. For a detailed discussion on multi-object trackers, see [14]. In this study, we propose a sequential (online) method that leverages information from a temporal window to jointly predict suitable tracks for each hand.

3. Experimental Setup

In this paper, we are interested tracking both hands of a driver in an unobtrusive yet reliable manner. To do so, we mount an over the shoulder, forward facing monocular RGB camera with an intent of capturing the whole region around the steering wheel, gear stick and instrument cluster (see Figure 1). Using this setup, we capture naturalistic video data of the subject driving. From this corpus of data, a total of 68 events are segmented, each of which capture the total or partial occlusion of one more hands of the driver. These events primarily correspond to the execution of right of left turn maneuvers as is shown in Figure 1. The 68 events collectively account for 2883 frames of video data, as well as left and right hand annotations corresponding to each frame. This dataset highlights a highly complex yet frequently occurring problem while tracking hands of a driver. This also makes it a very important challenge that must be overcome if tracking for long periods is desired.

4. Tracking Framework

The long-term object tracking problem is generally approached either from tracking or detection perspectives. Trackers generally use information from temporally adjacent frames to relocate objects in the current frame. However, trackers are prone to drift and fail when the object is either occluded or completely leaves the frame. Detection based algorithms estimate the object location in every frame independently. Detectors experience no drift and can detect objects that re-enter the frame. On the other hand, they may produce many false positives or fail to detect true neg-

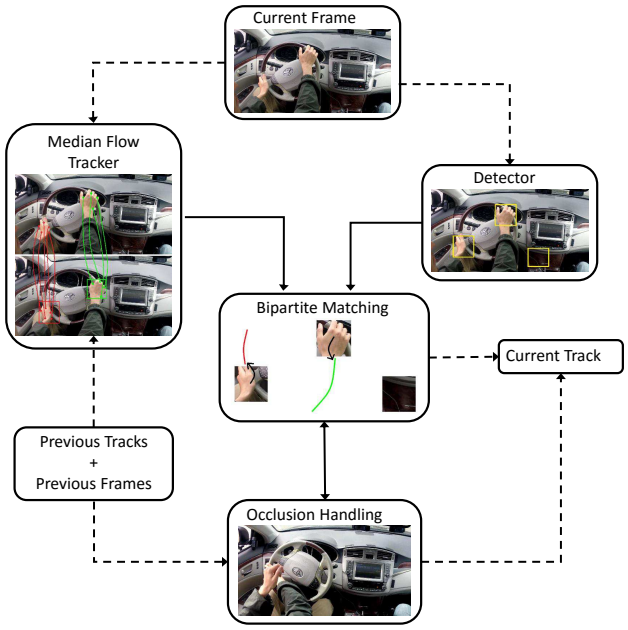


Figure 2: Block diagram of proposed tracking framework for a single frame. Dotted lines represent data flow and solid lines represent operation flow.

atives leading to an improper assignment of tracks. Hence, we propose to integrate the tracker and detector in a mutually beneficial manner to overcome each others' individual shortcomings. Figure 2 shows the block diagram of the proposed algorithm. The following subsections are devoted to explaining each individual block in detail.

4.1. Hand Detector

The detector is an integral part of our proposed framework. To make an astute choice for the detector, we experimented with different algorithms and feature selection schemes, taking special care to ensure that the detector is both complex enough to capture objects with high degree of freedom (such as hands), and fast enough to run in real (or near real) time.

First, we trained an ACF detector[6], carefully tuned for hand detection in the vehicle, using the VIVA hand detection dataset. The detector was trained using 10 channels (LUV + Normalized Gradient Magnitude + 6 x Gradient Orientation). The detector cascade consists of 4 stages of AdaBoost with 32, 128, 512, and 2048 weak learners for each stage respectively. The weak learner chosen was a depth 4 decision tree. A depth greater than 4 resulted in over-fitting. The template height was set at 65 pixels with an aspect ratio of 0.9.

Next, we trained a hand detector based on YOLO[20], a unified neural net based approach to object detection. A grid size of 15 and 2 bounding boxes per grid cell ensures that the network outputs a $15 \times 15 \times 11$ tensor of predictions. The grid size was increased from the original 7 to ensure better localization. The detector was first trained on images from the Oxford hand dataset[16], and then fine-tuned on the VIVA hand detection dataset. YOLO runs at roughly 25 fps for a 1280×720 video on a Titan X GPU.

Figure 3 depicts the PR curves of different detectors on both the VIVA challenge dataset, and the proposed hand occlusion dataset. YOLO is seen to significantly outperform the ACF detector in terms of overall performance. It is also seen to generalize well to new datasets as is seen from its performance on the occlusion dataset. Also note that the new dataset is significantly more challenging than the VIVA dataset, which is corroborated by the inferior AP/AR numbers obtained for each detector.

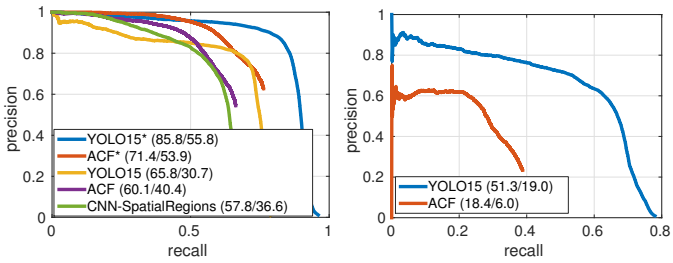
4.2. Median Flow Tracker

The hand detector alone is found to fall short of producing smooth object trajectories in challenging naturalistic driving settings. The detector fails to detect some hand instances, while introducing occasional false positives. We propose the use of a modified median flow tracker[9] to solve these issues. Given an set of bounding boxes from the previous frame, we initialize a set of keypoints[12] within each box to track in the next frame. The sparse motion flow of these keypoints are then determined using the pyramidal Lucas-Kanade algorithm[13]. Only points with a bi-directional (Forward-Backward) error less than 2 pixels are retained for the voting step. Additionally, points with a low skin likelihood are discarded to ensure the final track location remains on the hand. This also prevents keypoints that are not localized on the hand from dominating the voting procedure, thereby reducing the drift considerably. The median along both spatial dimensions (in the image plane) of all retained keypoints gives us the track location of the object for the given frame.

The median flow tracker may also be exploited to provide a bounding box estimate based on scale change. Scale change is computed as follows: for each pair of points, a ratio between current point distance and previous point distance is computed; bounding box scale change is defined as the median over these ratios. An implicit assumption of the point based representation is that the object is composed of small rigid patches. Parts of the objects that do not satisfy this assumption (object boundary, flexible parts) are not considered in the voting since they are rejected by the error measure. The bounding box is centered about the current track location.

4.3. Bipartite Matching for Data Association

Consider the problem of matching each of N tracks (from the) to one of M bounding box proposals. We formulate the



(a) PR curves for VIVA hand detection dataset.

(b) PR curves for proposed hand occlusion dataset.

Figure 3: Performance curves for the hand detectors on different datasets. YOLO15* and ACF* denote the PR curves for the detectors when the bounding box overlap criteria is reduced to 0.3.

probability of associating a track T_i, i = 1, 2, .., N with an object (bounding box) $O_j, j = 1, 2, .., M$ at a given instant of time as the product of three components (distance, detection, tracking):

$$P(O_j \in T_i) = P_{dist}(O_j \in T_i) P_{det}(O_j \in T_i) P_{trk}(O_j \in T_i) \quad (1)$$

Distance P_{dist} encodes the distance between the last know track location and the current object (bounding box) location

$$P_{dist}(O_j \in T_i) = \frac{d_{i,j^*}}{d_{i,j}}, \quad (2)$$

where

$$j^* = \underset{j}{\operatorname{argmin}}\, d_{i,j}, \ \forall j = 1, 2, .., M. \quad (3)$$

$d_{i,j}$ denotes the distance between the latest location of track T_i and the center of the object O_j.

Detection P_{det} ensures that false positives are weeded out before data association. It is defined as follows:

$$P_{det}(O_j \in T_i) = \begin{cases} 1 & \text{if score}(O_j) \geq 0.25, \\ 0 & \text{otherwise}, \end{cases} \quad (4)$$

where the score is the class probability obtained from the final layer of the neural network.

Tracking P_{trk} ensures that the bounding box proposals are reinforced by the tracking data. The score $P_{trk}(O_j \in T_i)$ is defined to be the fraction of keypoints (from the median flow tracker) associated to track T_i at the given time that is enclosed within bounding box O_j. This term reinforces belief in objects that are supported by the tracker, and devalues ones that are not.

Integrating three different cues while assigning probabilities helps handle the data association problem effectively by taking a more holistic view of the situation, thereby pushing the matching algorithm to reason over distance in consecutive frames, tracking data, and detection score confidence.

To find the optimal assignment between tracks and objects, we need to form a cost matrix $C = \{C_{i,j}\}$, with

$$C_{i,j} = -\log P(O_j \in T_i), \quad (5)$$

and then apply Hungarian algorithm to find the min-cost solution. If a finite cost solution is not found, the system updates each track using the median flow tracker.

5. Occlusion Handling

The tracking framework described above is seen to be capable of handling harsh illumination changes and rapid hand movements which frequently occur in naturalistic driving. However, self-occlusion is a recurrent event that the framework cannot seem to manage. After analyzing hours of real world driving data, this was seen to be a common phenomenon during turns, irrespective of the driver or the type of turn. To solve this problem effectively, we propose a data association mechanism that produces the most probable trajectory that each hand would follow, given a windowed history of their previous tracks.

The first step to handling occlusions is to detect such events. Indicators for such events include two tracks merging spatially, or when the total cost of data association exceeds a chosen threshold for a contiguous set of frames.

5.1. Track Approximation

We then proceed to encode the windowed history corresponding to both tracks. A track T consists of N pairs of xy-positions in the $2D$ plane together with associated timestamps t_i as below:

$$T = (((x_0, y_0), t_0), ..., ((x_{N-1}, y_{N-1}), t_{N-1})), \quad (6)$$

where $t_i < t_{i+1}$ for $i = 0, ..., N-1$. The track T can be restricted by several constraints, e.g., by the number of elements, the distance or a time constraint. In both latter cases, the number of elements N can vary because it is not guaranteed that any of the measured values are equidistant in time. In our application, we consider a window size that captures the length of the latest tracklet corresponding to the object. A *tracklet* in this context is a set of spatially overlapping track locations. This is large enough to ensure that significant hand motion corresponding to the maneuver is captured. To get a uniform representation of an arbitrary length track, a Chebyshev decomposition on the components of the

track (xy-positions in the image plane) is applied. This representation also reliably encodes the motion history of an object in a succinct form. The coefficients of this polynomial approximation are used as input features of the prediction model. The Chebyshev polynomial T_n of degree n is defined by

$$T_n(x) = cos(n\ arccos(x)), \tag{7}$$

which looks trigonometric but can be shown to be polynomials. The first two polynomials are defined by

$$T_0(x) = 1 \tag{8}$$

and

$$T_1(x) = x. \tag{9}$$

Using the recursive formula

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \forall n \geq 1, \tag{10}$$

one can easily calculate the polynomials of higher order. To approximate an arbitrary function $f(x)$ in the interval $[-1, 1]$, the Chebyshev coefficients are defined by

$$c_n = \frac{2}{N} \sum_{k=1}^{N-1} f(x_k) T_n(x_k), \tag{11}$$

where x_k are the N zeros of $T_N(x)$. The reconstruction formula is defined as

$$f(x) \approx \sum_{n=0}^{m-1} c_n T_n(x) - \frac{1}{2} c_0, \tag{12}$$

where $m \leq N$ can be used to control the approximation quality. For a detailed description on the Chebyshev approximation and its advantages compared to other approximations for spatio-temporal trajectory modeling, see[19, 4].

Both xy-components of each track are transformed to the interval $[-1, 1]$ and the Chebyshev decomposition is applied. This results in two m-dimensional vectors, $\mathbf{c_x}$ and $\mathbf{c_y}$, of approximation coefficients, one for each of x and y. The final feature vector is formed through a simple concatenation as follow:

$$\mathbf{v} = [\mathbf{c_{x,l}}, \mathbf{c_{y,l}}, \mathbf{c_{x,r}}, \mathbf{c_{x,r}}] \in \mathcal{R}^{4m}, \tag{13}$$

where the supplementary subscripts **l** and **r** denote if the approximation coefficients belong to the left or right hand respectively. The number of coefficients used influences the approximation quality. Experiments showed that $m = 5$ coefficients are enough to obtain a good approximation of the tracks.

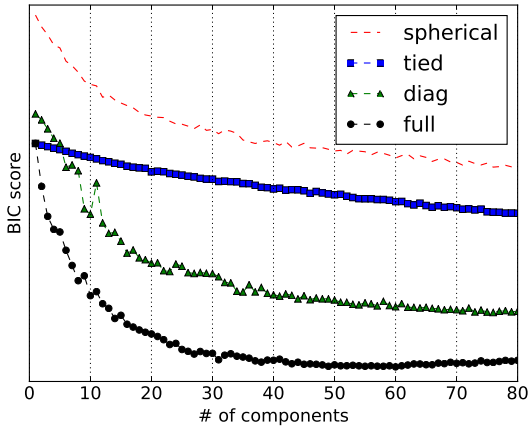


Figure 4: BIC scores as a function of number of components in GMM.

5.2. Matching Past and Future Tracks

First, we start off by extracting tracks leading to and from occlusion events from a set of training video sequences. Let t_o denote the time index corresponding to the onset of an occlusion event in a video sequence. Let T_i denote the track of an object before time index t_o, and T_o denote the track of the same object after. The temporal lengths of T_i and T_o may be chosen arbitrarily as long they ensure that T_i begins and T_o ends when both hands are completely out of occlusion. Next, we sample contiguous segments with length greater than or equal to 5 from T_i and T_o to generate sets $\mathbf{T_i}$ and $\mathbf{T_o}$ respectively. Let $\mathbf{T_{i,l}}$ and $\mathbf{T_{i,r}}$ denote the set of incoming segments for the left and right hand respectively. Similarly, let $\mathbf{T_{o,l}}$ and $\mathbf{T_{o,r}}$ denote the set of outgoing segments for the left and right hand respectively. Note that there is a one-to-one correspondence between each element in $\mathbf{T_{i,l}}$ and $\mathbf{T_{i,r}}$, and between each element in $\mathbf{T_{o,l}}$ and $\mathbf{T_{o,r}}$. For each such correspondence, we generate a vector of Chebyshev coefficients using (13). This gives us sets of coefficients $\mathbf{C_i}$ and $\mathbf{C_o}$ respectively obtained from encoding ($\mathbf{T_{i,l}}$, $\mathbf{T_{i,r}}$) and ($\mathbf{T_{o,l}}$, $\mathbf{T_{o,r}}$) respectively. The final set of features for the given occlusion event is generated as:

$$\mathbf{X_j} = \mathbf{C_i} \times \mathbf{C_o}, \tag{14}$$

where $\times$ denotes the Cartesian product between sets. The final set of features from the entire training corpus is simply:

$$\mathbf{X} = \bigcup_j \mathbf{X_j}. \tag{15}$$

Note that each feature $\mathbf{x} \in \mathbf{X}$ is a concatenation of two coefficient vectors $[\mathbf{v_i}, \mathbf{v_o}]$ where $\mathbf{v_i} \in \mathbf{C_i}$ and $\mathbf{v_o} \in \mathbf{C_o}$.

To match past and future tracks during occlusion events, we learn a Gaussian Mixture Model (GMM) from the feature set $\mathbf{X}$. To choose the optimal number of components in the mixture, we use the Bayesian Information Criterion (BIC) as a measure of data fit to the model. We plot the BIC scores as a function of the number of components in the mixture for different covariance types (see Figure 4). The model corresponding to the lowest score (full covariance with 48 components) is chosen for further experiments.

During operation, when an occlusion is detected, we first extract the vector of coefficients $\mathbf{v_i}$ from the latest tracklet associated with each object. Once this is done, the tracker goes into a state of *dormancy* until both hands are completely out of occlusion. This event may be detected when atleast two strong tracklets of a desired length are observed. Once this condition is satisfied, we generate a set of proposals for the vector of output coefficients $\{\mathbf{v_{o,j}}\}$ obtained from each pairwise combination of outgoing tracklets. The optimal outgoing vector of coefficients is then chosen as follows:

$$\begin{aligned} \mathbf{v_o^*} &= \underset{\mathbf{v_{o,j}}}{\operatorname{argmax}} P(\mathbf{v_{o,j}} \mid \mathbf{v_i}) \\ &= \underset{\mathbf{v_{o,j}}}{\operatorname{argmax}} P(\mathbf{v_i}, \mathbf{v_{o,j}}) \end{aligned} \tag{16}$$

where $P(\mathbf{v_i}, \mathbf{v_{o,j}})$ is the GMM trained using $\mathbf{X}$.

This solves the problem of associating incoming and outgoing tracklets corresponding to each object after occlusion since $\mathbf{v_o^*}$ includes coefficients corresponding to both left and right hands. Intuitively, the GMM models this association of tracklets corresponding to different hand motions encountered while driving. For a given set of maneuvers, it is seen that hand motion patterns tend to repeat themselves, thereby enabling tracklet association to be learned offline.

5.3. Interpolation

Once the incoming and outgoing tracklets corresponding to each object are associated, the tracking continues as normal. However, one may still be interested in recovering the tracks of objects during occlusion. This can be done by concatenating each pair of associated tracklets, finding the corresponding coeffiecients using (11), and then unpacking the coefficients to the desired length of the interpolation window using the reconstruction formula in (12).

6. Experimental Evaluation

In order to evaluate the proposed framework, the occlusion dataset is split into two parts for training and testing. We use 46 video sequences for training, and the remaining 22 for testing. We ensure that the test dataset consists of all hand gestures that correspond to frequently performed drive maneuvers. We also include maneuvers that result in frequent self-occlusion of the hands to thoroughly test the occlusion handling capabilities of our proposal. The dataset mimics real world conditions and offers challenges such as different subjects, different capture settings, background clutter and harsh illumination.

We use the standard CLEAR-MOT[2] metrics to evaluate our proposal (**TD + OH**). A state of the art baseline: **Discrete-Continuous Energy Maximization (DC-EM**[15], is chosen for comparison and evaluation. All hyper-parameters are tuned using the training dataset. It must be noted that **DC-EM** is an offline method and thus predicts tracks after observing the entire video sequence as a whole. In addition to this, we also include the results for the proposed framework without occlusion handling capabilities (**TD**). This gives us a tangible sense of the effect that occlusion handling has on the overall performance of the tracker. All trackers make use of the same detection box proposals obtained from the YOLO based hand detector.

As can be seen in Table 1, the tracker without occlusion handling (**TD**) performs poorly in comparison to the other two techniques. This is expected as it performs satisfactorily only under normal conditions. This makes it unsuitable for long term tracking. The baseline (**DC-EM**) provides much better results in comparison to (**TD**). This is because it is capable of handling occlusions and missing tracks *a posteriori* after observing the entire sequence. It is also seen that integrating the occlusion handling module (**TD + OH**) produces large gains in overall performance. Crucially, the occlusion handling module also reduces the number of ID switches considerably, thereby demonstrating better tracklet association. This module enables maintaining reliable tracks for a longer duration, while handling all common hand gestures in an online fashion. Figure 5 shows example results for tracking through occlusion events. These results indicate that common hand gestures learned offline, generalize well enough for prediction purposes. This idea may be extended to other applications where the subject carries out a repeated set of hand gestures that can be reproduced reliably.

7. Concluding Remarks

This paper introduces a novel tracking framework designed for long-term analysis of vehicle occupants' hand movements. The uniqueness and benefits of such a tracker in comparison to a generic MOT are highlighted, and a case is made for its separate consideration. A combined tracking and detection framework is proposed to produce individual tracks online, and data association is performed using the Hungarian algorithm. Although motion and appearance-based tracking is adopted, these alone provide difficult disambiguation when the hands are occluded or interacting. Therefore, we introduce a module to handle such cases by encoding hand motion patterns offline. The proposed algo-

Table 1: Results on test dataset for different tracking schemes. Arrows next to each metric indicate if a higher(↑) or lower(↓) is desired.

Method	MOTA (↑)	MOTP (↓)	MT (↑)	ML (↓)	IDS (↓)	Frag (↓)
TD + OH	0.650680	0.753181	0.777275	0.000000	4	30
TD	0.233710	0.626504	0.000000	0.550000	16	83
DC-EM	0.535302	0.588956	0.340909	0.113636	48	79

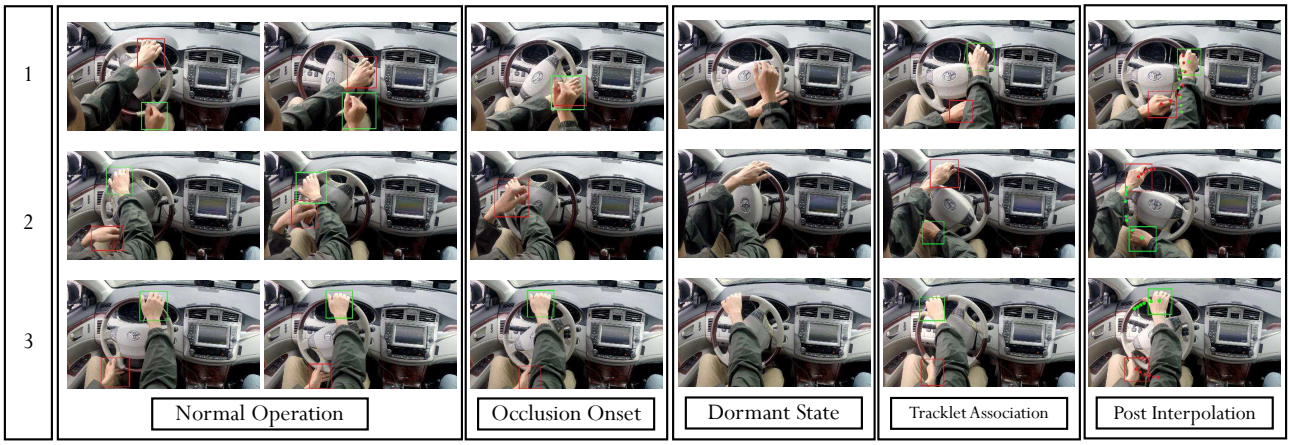


Figure 5: Example results from the proposed framework for three occlusion sequences. Different stages of operation of the tracker are highlighted for better comprehension. The tracks for the left (red) and right (green) hand are color coded for convenience.

rithm shows significant improvement over a state of the art baseline, with occlusion handling alone accounting for large gains. This approach to managing occlusions may easily be carried over to other applications, where objects of interest tend to produce similar motion patterns in the long run.

References

[1] A. A. Argyros and M. I. Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. In *Computer Vision-ECCV 2004*, pages 368–379. Springer, 2004.

[2] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *Journal on Image and Video Processing*, 2008:1, 2008.

[3] M. Breig and M. Kohler. *Motion detection and tracking under constraint of pan tilt cameras for vision based human computer interaction*. Citeseer, 1998.

[4] Y. Cai and R. Ng. Indexing spatio-temporal trajectories with chebyshev polynomials. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 599–610. ACM, 2004.

[5] S. Y. Cheng, S. Park, and M. M. Trivedi. Multi-spectral and multi-perspective video arrays for driver body tracking and activity analysis. *Computer Vision and Image Understanding*, 106(2):245–257, 2007.

[6] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(8):1532–1545, 2014.

[7] A. Doshi and M. M. Trivedi. Tactical driver behavior prediction and intent inference: A review. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1892–1897. IEEE, 2011.

[8] A. Gaidon and E. Vig. Online domain adaptation for multi-object tracking. *arXiv preprint arXiv:1508.00776*, 2015.

[9] Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-backward error: Automatic detection of tracking failures. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 2756–2759. IEEE, 2010.

[10] Z. Khan, T. Balch, and F. Dellaert. Mcmc-based particle filtering for tracking a variable number of interacting targets. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1805–1819, 2005.

[11] S. G. Klauer, F. Guo, J. Sudweeks, and T. A. Dingus. An analysis of driver inattention using a case-crossover approach on 100-car data: Final report. Technical report, 2010.

[12] M. Kölsch and M. Turk. Flocks of features for tracking articulated objects. In *Real-Time Vision for Human-Computer Interaction*, pages 67–83. Springer, 2005.

[13] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

[14] W. Luo, J. Xing, X. Zhang, X. Zhao, and T.-K. Kim. Multiple object tracking: A literature review. *arXiv preprint arXiv:1409.7618*, 2014.

[15] A. Milan, K. Schindler, and S. Roth. Multi-target tracking by discrete-continuous energy minimization. *Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[16] A. Mittal, A. Zisserman, and P. H. S. Torr. Hand detection using multiple proposals. In *British Machine Vision Conference*, 2011.

[17] E. Ohn-Bar, A. Tawari, S. Martin, and M. M. Trivedi. Predicting driver maneuvers by learning holistic features. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pages 719–724. IEEE, 2014.

[18] E. Ohn-Bar and M. Trivedi. Joint angles similarities and hog2 for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 465–470, 2013.

[19] W. H. Press. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015.

[21] B. Stenger, P. R. Mendonça, and R. Cipolla. Model-based 3D tracking of an articulated hand. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–310. IEEE, 2001.

[22] Computer Vision and Robotics Research Laboratory, UCSD. Vision for Intelligent Vehicles and Applications (VIVA). `http://cvrr.ucsd.edu/vivachallenge/`, 2015.

[23] J. Tison, N. Chaudhary, and L. Cosgrove. National phone survey on distracted driving attitudes and behaviors. Technical report, 2011.

[24] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon. Bayesian multi-object tracking using motion context from multiple objects. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 33–40. IEEE, 2015.