Automatic Refinement Strategies for Manual Initialization of Object Trackers

Hao Zhu, *Student Member, IEEE,* Fatih Porikli, *Fellow, IEEE*

***Abstract*—Tracking objects across multiple frames is a well-investigated problem in computer vision. The majority of the existing algorithms assume an accurate initialization is readily available. However, in many real-life settings, in particular for applications where the video is streaming in real-time, the initialization has to be provided by a human operator. This limitation raises an inevitable uncertainty issue.**

Here, we first collect a large and new dataset of inputs that consists of more than 20K human initialization clicks, called as HIC, by several subjects under three practical user interface scenarios for the popular TB50 tracking benchmark. We analyze the factors and mechanisms of human input, derive statistical models, and show that human input always contains deviations, which exacerbate further when the relative object-camera motion becomes large. We also design and evaluate alternative refinement schemes, and propose a strategy that refits an object window on the most probable target region after a single click. To compensate for the human initialization errors, our method generates window proposals using objectness cues extracted from color and motion attributes, accumulates them into a likelihood map that is weighted by the initial click position and visual saliency scores, and assigns the final window by the maximum likelihood estimate. Our experiments demonstrate that the presented refinement strategy effectively reduces human input errors.

***Index Terms*—Object initialization, object tracking, human-computer interactive, error compensation.**

I. Introduction

VISUAL tracking plays an important role in computer vision applications. In the past, a vast number of approaches have been developed to address various tracking challenges, such as illumination and appearance changes, full and partial occlusions, background clutter, uncontrolled camera motion, and low contrast [1], [2], [3], [4], [5], [6], [7], [8], [9]. Most of the existing tracking methods presume that the given initialization information, which is usually a rectangle box in the first frame, is correct and sufficient enough to launch the tracking process. Nevertheless, this strong assumption often causes the state-of-the-art trackers to suffer from degraded performance due to inaccurate priors [7], [10], [11], [12].

It is unlikely that a precise initialization to be available for all uses. For applications that operate with a wide spectrum of objects, learning accurate models for automatic initialization is not a feasible option, thus the initial target location for tracking has to be provided by an expert. Many real-life human-computer interaction scenarios, such as selective-focus in digital cameras and target tracking in robot navigation, are designed to work for a general class of objects with manual initialization under different circumstances. As a result, the accuracy of the human input becomes a critical factor.

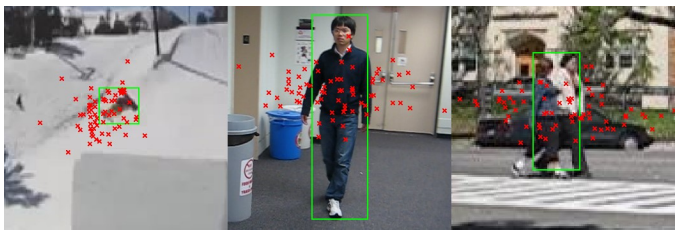


Fig. 1. Human initialization click distributions for different videos. Since the human reaction times vary due to many factors, input can be in any frame. As visible, when the clicks are entered, target objects may have already moved significantly, resulting in inaccurate clicks outside the object regions.

Intuitively, initializing the target with more meticulous information, such as object boundaries and intricate models, can reduce the uncertainty and minimize tracking drift issues. However, more complex information require longer user interaction time. This is not possible for streaming video and moving camera applications, for instance, deployed on a handheld device or a pan-tilt-zoom surveillance camera. Coupled with such restrictive operation conditions and real-time processing requirements, most tracking applications can only support simple and quick initialization schemes. This invokes the demand for inferring not only the center location but also the scale and object boundary from a single click input.

In this paper, we aim to provide a deeper understanding to the limitations of human initialization and establish a human input model. To this end, we use a computer interface that allows human subjects to click on targets while a video sequence is being displayed. We collect and analyze several aspects of a large corpus of actual human initialization clicks, HIC dataset, captured for one of the most extensive object tracking benchmarks, TB50 [13].

The HIC dataset is the first and only publicly available dataset that contains instances of human provided initializations. Here, a click is an entry that represents one (or a few) image coordinates. Albeit simple and quick, a single click is not fully reliable. Due to many elements including erratic camera and object motion patterns, latency in human reaction times, small object size, and system interface impediments,

The work of H. Zhu was supported in part by the National Natural Science Foundation of China under Grant 61321002 and Grant 61120106010, and in part by the China Scholarship Council (No.:201406030023). The work of F. Porikli was supported under the Australian Research Councils Discovery Projects funding scheme (project DP150104645).

H. Zhu is with the School of Automation, Beijing Institute of Technology, and also with the State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing 100081, China. Email: richie.zhu08@gmail.com.

F. Porikli is with Australian National University, Canberra, ACT 0200, Australia. E-mail: fatih.porikli@gmail.com

The first author H. Zhu and the second author F. Porikli have equal contributions.

it is difficult to ensure the consistency and accuracy of the human input. As can be seen in Fig. 1, click positions contain significant deviations from their ground-truth locations due to object-camera motion. These deviations may consequently cause degeneration even at the very beginning of the tracking process. Therefore, it is necessary to refine the click positions to improve the input accuracy.

Aiming to fill the gap between what a click provides and what degree of reliability in the initialization step a tracking algorithm requires, we focus on a refinement process that can

- infer object region from a single click for arbitrary targets;
- robust to relative camera-target motion, existence of multiple objects, and scene clutter;
- reduce human input errors; and,
- be independent from the tracker.

To minimize the adverse effects of unreliable initializations, our second contribution is a multi-cue based refinement framework that repositions the click and estimates the target scale for each human click. This refinement is based on the maximum likelihood estimate using an *objectness* confidence map, which combines both saliency cue and initial click information. We conduct extensive experiments with different component selection options to determine an optimal combination. In the case of the existence of multiple objects, the refinement process relies more on the human clicks, and we make further analysis of this scenario in the experimental section.

The remainder of this paper is organized as follows: We present an overview of the related work in Section II. Our HIC click modeling is explained in detail in Section III. The refinement process is described in Section IV. Our experimental evaluation and results are presented in Section V.

II. Related Work

A comprehensive overview of the existing tracking methods is outside the scope of this paper. Here, we discuss the tracking approaches in the light of their robustness to initialization errors.

Several remarkable object tracking benchmarks [10], [11], [12], [14], [15] have evaluated and compared popular trackers. Experimental protocols of some benchmarks also advocate initializing trackers with perturbed bounding boxes to analyze the robustness to initialization errors. For instance, the protocol in [11] uses spatial translations (10% of target size) and scale variations (0.8×-1.2×) to generate alternative initializations. The reported results for [11] show that the bounding boxes with larger scale often lead to significant decrease in performance since larger regions contaminate initial window with more background pixels and many of the state-of-the-art trackers, such as [5], [16], [17], are sensitive to the subversion of the object model. For the same purpose, the VOT challenge [12] adds translations (10% of the target size) and rotations ($\pm$0.1 radians) to the ground-truth boxes. Although there is no scale perturbation and the translation is rather small, majority of the tested trackers still suffer significant performance degradation. The noisy initialization experiment in [10] contains more severe spatial errors up to 20% displacements of the initial window, which consequently, causes an average performance loss of 10% for all trackers.

All these extensive studies indicate that even initialization errors less than 20% deteriorate the performance. Our experiments exposed that the typical error in human clicks could be even much larger.

Few trackers considered handling inaccurate initializations. Among those, the *initialization-insensitive tracker* (IIT) [7] is based on the generalized Hough transform voting [18] weighted by both motion similarity and descriptor similarity of feature points. To cope with imperfect initializations, it learns a feature set and updates the set in an online fashion, however, the model is still depends highly on the bounding box in the first frame. Therefore, an initialization with significant deviation still distracts IIT. Besides, the benchmark results in [12] show that while IIT focuses on initialization robustness, its general performance falls behind the conventional trackers.

Although most tracking methods do not have specific mechanisms to deal with inaccurate initializations, some react more robustly than the others [10]. For instance, *Struck* [2] benefits from maintaining a positive and negative set of support vectors sampled according to an overlap based similarity measure in its classifier [10] and Haar-like features that are less sensitive to background clutter [11]. *Hough-based tracker* [6] makes a new segmentation using GrabCut [19] based on a rough classification in every new frame. The segmentation process helps to reduce the negative effects of inaccurate input, yet the initial classifier still relies heavily on the initial information. In [20], a space containing a set of trackers in different states is maintained. This method gets samples from the space and constructs the most fitting combination during the tracking process. Strong subsets of the tracker states make it robust to slight errors. However, similar with *Struck* and *Hough-based tracker*, the robustness against initial disturbances is still restricted within a relatively small range.

Instead of starting the tracking process with a given window in the first frame, some vision systems employ automatic initialization schemes utilizing motion cues such as background subtraction [21], [22], long-term trajectory consensus [23], [24], motion edge [25], and contours [26]. Nevertheless, object movements are not always immediately available at the initialization stage, which impedes motion indicators. Besides, analyzing long-term trajectories requires abundant memory and computational resources [27], [28], thus, these schemes are not suitable for most applications. In case a model of the target object is available before tracking, the initialization can be achieved by a detector as in [29], [30], [31], [32]. Obviously, these methods are not applicable to arbitrary objects. Saliency is another cue considered for tracking and automatic initialization [33], [34]. A bottom-up saliency map can suppress background regions and highlight salient regions. Yet, there is no guarantee for the salient region to correspond to the target object [35]. Due to these serious restrictions of automatic initialization schemes, manual initialization process is widely used for most real-world applications.

III. Modeling of Click Distribution

There is no question that the target object can be marked accurately if we can stop the video stream for a sufficiently long duration. However, this may not be possible for many reasons:

- For streaming video, one can pause video only for a duration that the processing time of the buffered frames can be absorbed back quickly to prevent from persistent latency. Considering the typical computational speeds of the state-of-the-art trackers and memory limitations of processing platforms, the latency due to buffering cannot be neglected. Dropping frames may not be an option either as the increased relative motion of the target may lead to negative effects on trackers.
- In real-time tracking tasks, e.g. selective zoom feature for handheld devices, there exist stringent reaction time requirements subjective to the motion of the object and camera.
- In tracking-driven applications, e.g. PTZ camera systems where camera movement is directly dictated by tracking results, one cannot simply stop the camera, thus pause video, for long due to the mechanical limitations of the system.

To evaluate realistic initialization operations, we build a visual interface to collect clicks from different human subjects. Based on the interface, we design three scenarios that simulate different system constraints. From mild to severe, they are *Pause&Click*, *Follow&Click*, and *Spot&Click*.

- **Pause&Click:** In this scenario, subjects are allowed to temporally pause the video stream at any time and then click on the target center in the paused frame. The time cost is defined as the duration between pausing and clicking. While subjects are advised to click as soon and as accurate as possible, they have the freedom of balancing the reaction time and the clicking precision.
- **Follow&Click:** In the second scenario, the pause option is unavailable. As we do not impose a constraint about when to click, users are only asked to click on the target center as accurately as possible. They are allowed to visually observe the target object and make their entry anytime (any frame) they are comfortable.
- **Spot&Click:** The last scenario is more challenging as it sets a reaction time constraint on the user in addition to not having the capability to pause the video. Subjects are required to click on the target object within a maximum time limit, which is 1.5 seconds in our experiments.

In the following parts of this section, we describe the experimental settings of the click collection (Section III-A and III-B), analyze the *Pause&Click* and other two scenarios (Section III-C), and discuss the click models (Section III-D) in detail.

A. Object Tracking Dataset

We use a comprehensive tracking benchmark dataset, TB50 [11], to collect human clicks. TB50 contains 49 sequences with 26,529 frames covering various target classes such as faces, pedestrians, different animals, vehicles, and other rigid and nonrigid objects. The scenes include indoors, outdoors, daytime, and nighttime. These videos are captured from TV broadcasts, surveillance cameras, and handheld devices (see samples in Fig. 8). The original frame size varies from 128×96 to 768×576. The frame rate of the videos is 25 fps.

To categorize videos according to their target attributes, we use the average target velocity and target size of each video sequence. These attributes are computed only within the parts of the videos where the human subjects are given the task of clicking on the target object in order to allow an objective comparison across different videos. The velocity and size are based on normalized frames (short-sides to a fixed value).

B. Human Initialization Click (HIC) Collection

Marking object area with a rectangle or drawing object boundary requires significant user interaction, which makes them unsuitable for time-critical applications. Therefore, we use much faster single click inputs albeit no scale information is provided by a single click.

On the TB50 dataset, we collected more than 20,000 tracking initialization clicks from 10 skilled human operators. We presented these subjects a short demonstration of initialization procedures and ground-truth object windows in order to make them familiar with the interface and each scenario. We asked the subjects to exercise clicking action on sequences that are not in the TB50 dataset before collecting the initialization clicks.

During the click collection, we imposed these guidelines:

- To simulate diversity in object appearance, a random initial frame used for each sequence at each click.
- To prevent subject bias, the same number of clicks are accepted from each subject for each sequence.
- To minimize memorization bias that may happen due to watching the same sequence, only a small number of clicks per sequence from each user is collected.
- Unintentional and clicks on the wrong objects are removed.

On the visual interface, we normalize all videos to the same size (short-sides to a fixed value) while keeping their aspect ratios constant. This is aimed to have a consistent visual field of attention. In case the video frames are displayed in their original size on the screen, larger frames may involve more time to move the input device, e.g. mouse, thus more time to click. The average target size for the dataset after the display normalization is 37×53 and the average frame size is 367×243.

C. Click Accuracy

To measure the accuracy of the clicks, we use the spatial distance between click position and ground-truth object center. Target objects in different sequences are different in size, hence the spatial error cannot be directly compared. For example, a click with 15-pixels error stays inside a 40×40 target, however, it is outside a 10×10 target. Therefore, for accuracy assessments we normalize the target window to a 50×50 unit region in every frame for comparisons.

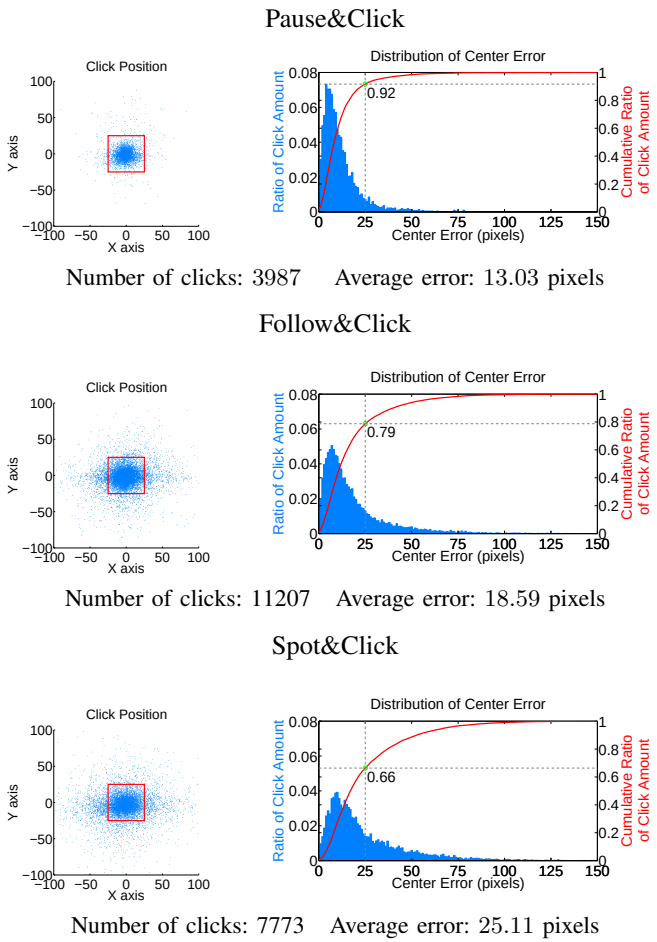


Fig. 2. HIC distributions for the whole TB50 dataset. Rows show *Pause&Click*, *Follow&Click* and *Spot&Click* scenarios, respectively. Target windows are normalized to unit size (50×50). Best viewed on screen and in color.

We also analyze the overlap ratio between the ground-truth region and a window centered on the click. The size of the window is set to the ground-truth object size. Notice that, no object size information is available in-situ, thus the actual overlap might be much lower.

The statistics of human clicks are given in Table I. As visible, the initialization error increases along with the operational restrictions.

For the *Pause&Click* scenario, the subjects take an average 14.84 frames for clicking on the target objects with an average distance error of 13.03 pixels with respect to unit region. We interpret this error as due to the instruction of quick reaction time as well as the natural variance between human operators who supplied ground-truth information and clicks. At 25 fps, this corresponds to an average target movement of 63.10 pixels (in normalized frames), meaning that the target center is often outside the original target region when the click is provided.

For the *Follow&Click* and *Spot&Click* scenarios, the HIC distributions are shown in Fig. 2. The red rectangle indicates the normalized ground-truth region and each blue point is a click. As can be seen, there are many clicks having relatively large errors.

For the *Follow&Click*, the average error for all video sequences (including the erratic motion videos) is 18.59 pixels and 19.55% of the clicks are located outside the ground-truth target region. These numbers increases to 25.11 pixels and 30.45% respectively for the *Spot&Click*.

The best overlap performance between the ground-truth window and the window with click center and the correct object scale is only around 0.63, which is for the smooth and slow motion sequences. The overlap dramatically decreases for the erratic motion sequences. This shows more severe initial distortions than the experimental protocols provided by the benchmarks [10], [11], [12]. As a result, typical error is expected to be much larger indicating the practical tracking applications faces with more challenging situations when user initializes the target object.

D. Influence of Target Velocity and Target Size

There is significant error diversity across different sequences. To determine what leads to this variance, we focus on two attributes: *target velocity* and *target size*.

Figure 3 shows the sequences ranked according to their target velocity and size. As expected, the click error has a positive correlation with the target velocity. Faster targets, in particular those with erratic motion, cause larger errors. For the category of sequences containing fast and erratic camera-target motion, such as the ones captured by handheld cameras, even the most careful subjects may fail to initialize the targets accurately. HIC distributions of the smooth and erratic motion sequences are given in Fig. 4. As shown, the error for the erratic motion sequences is higher than the smooth motion. Under the *Pause&Click* scenario, the average center error increases from 11.75 pixels for smooth motion to 17.29 pixels for erratic motion. For more realistic *Follow&Click* and *Spot&Click* scenarios, the increase is from 14.72 pixels to 32.63 pixels, and from 22.01 pixels to 35.73 pixels, respectively. As presented in Table II, approximately 16%, 51%, and 56% of the clicks are outside the ground-truth regions for the *Pause&Click*, *Follow&Click*, and *Spot&Click* in case of the erratic motion. These numbers drastically drops to 5%, 11%, and 24% in case of smooth motion.

TABLE II
RATIO OF OUT-REGION CLICKS

	All	Smooth	Erratic
Pause&Click	8% 307 / 3987	5% 157 / 3065	16% 150 / 922
Follow&Click	19.55% 2191 / 11207	11% 1003 / 8788	51% 1230 / 2419
Spot&Click	30.45% 2369 / 7773	24% 1470 / 6016	56% 984 / 1757

The click accuracy is negatively correlated with the target size. When we ranked the tracking sequences according to their *target size* (see Fig. 3), we observe that the smaller targets lead to less accurate human input. This negative influence becomes more significant for the *Follow&Click* and *Spot&Click* scenarios since small targets are difficult to follow and spot under more stringent operating conditions.

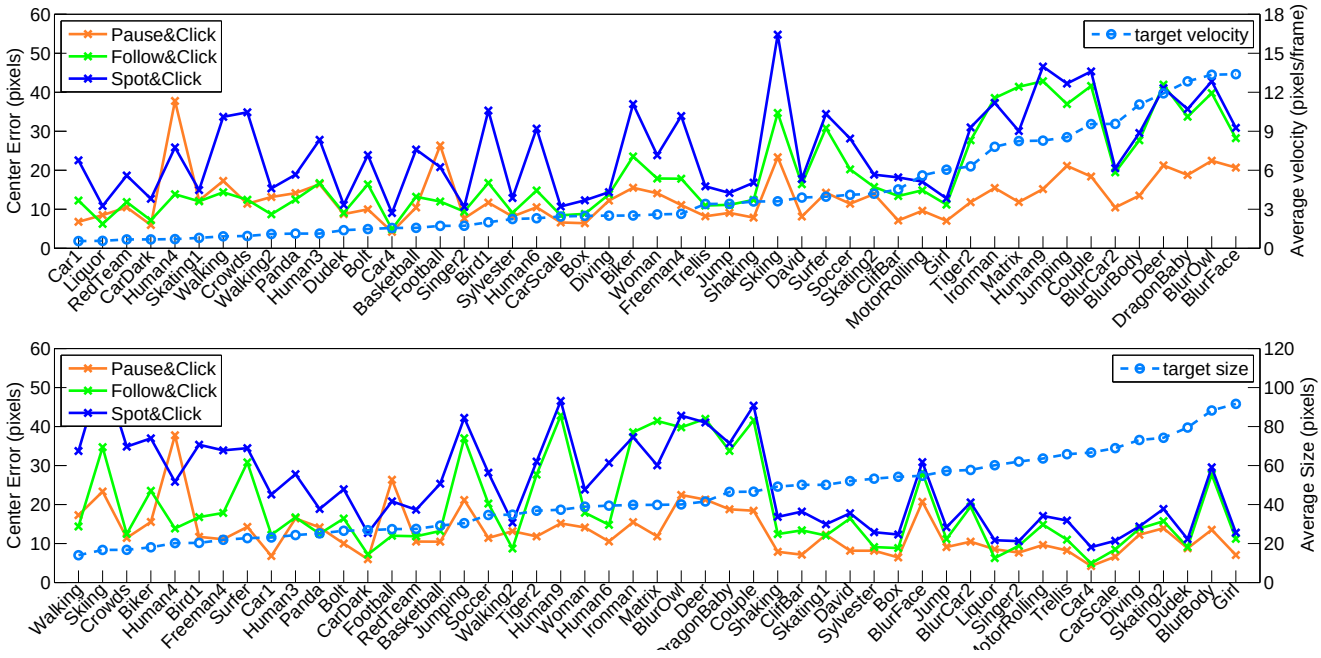


Fig. 3. Average center distance error ranked by the target velocity and target size respectively. The center errors are based on the normalization that all targets are resized to a 50×50 unit region while the target velocity and target size are consistent to users' observation.

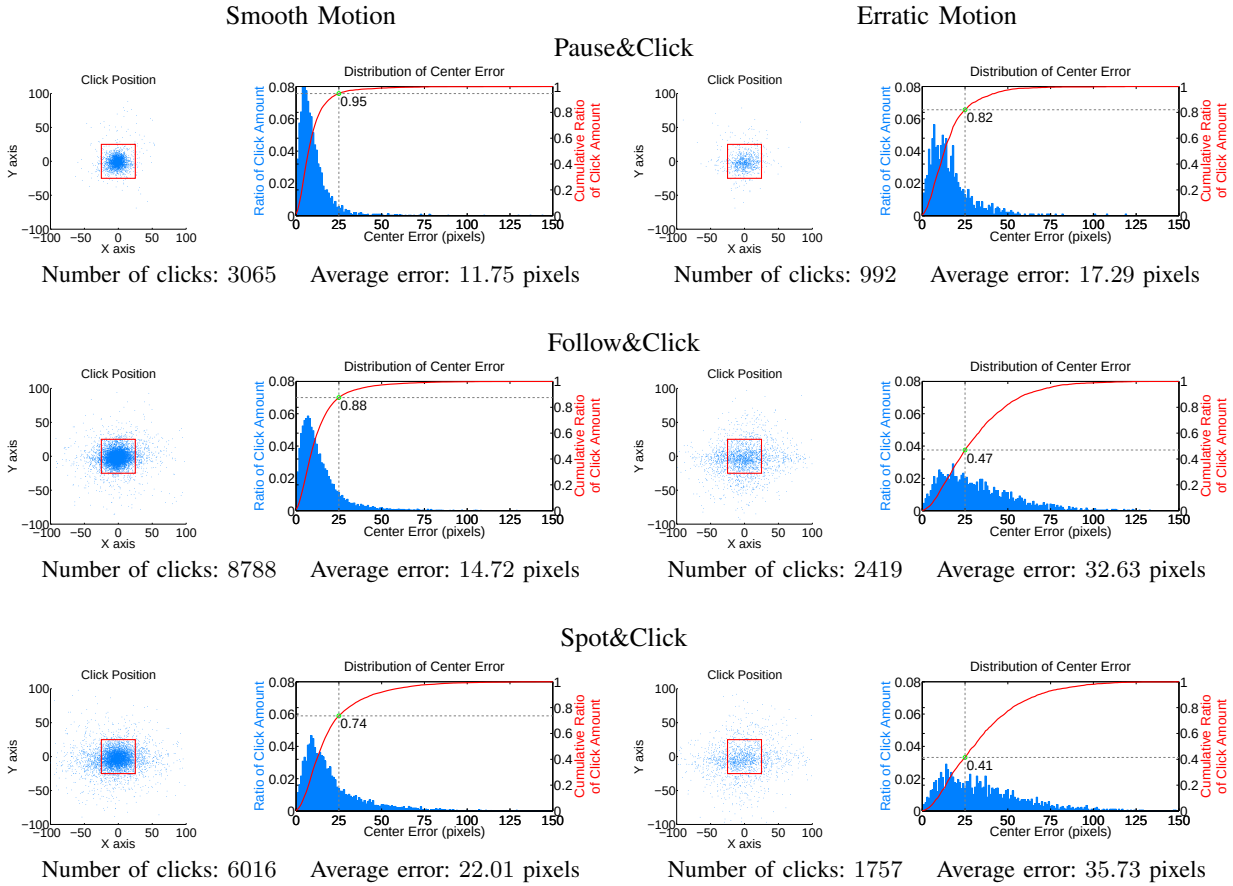


Fig. 4. HIC distributions for smooth (left) and erratic (right) motion sequences. The click accuracy deteriorates when motion becomes more unpredictable.

TABLE I
STATISTICS OF HUMAN INITIALIZATION CLICKS

Scenario	Sequence	Clicks	Distance Error		Overlap	
			Mean	Variance	Mean	Variance
Pause&Click	smooth motion	3065	11.75	285.41	0.63	0.04
	all	3987	13.03	267.68	0.60	0.05
	erratic motion	922	17.29	185.17	0.49	0.05
Follow&Click	smooth motion	8788	14.72	184.94	0.55	0.05
	all	11207	18.59	307.48	0.50	0.07
	erratic motion	2419	32.63	501.37	0.29	0.06
Spot&Click	smooth motion	6016	22.01	408.04	0.44	0.06
	all	7773	25.11	480.98	0.40	0.07
	erratic motion	1757	35.73	585.00	0.26	0.06

IV. AUTOMATIC REFINEMENT PIPELINE

Above, we showed that initialization errors are inevitable and can dislocate the initial click substantially outside the correct target region. Since most object tracking algorithms require sufficiently accurate initial object region, here we present a multi-cue based refinement method to compensate for the initialization errors.

Our refinement method is designed for arbitrary sized and general class of targets, therefore it does not require any other prior information or pretrained models. The pipeline of proposed framework is illustrated in Fig. 5. We formulate the task of finding the refined position as a maximum likelihood estimate. To determine the most likely object region, we employ a set of objectness and saliency cues in addition to the click information, compute a target likelihood map that aggregates multiple confidence maps of all candidate object windows, and select the window that has the highest likelihood score.

More specifically, we select the window B^* that has the maximum response on the target likelihood map $L(B_i;\mathbf{c})$

$$B^* = \arg\max_{B_i} L(B_i;\mathbf{c}), \qquad i = 1, ..., N \tag{1}$$

over N candidate windows B_i generated by an object proposal method, for instance, MCG[36], EdgeBox[37], BING[38], RPN[39] using color and motion cues, and considering the initial click position $\mathbf{c} = [x_c, y_c]$.

Assuming a mixture model, the likelihood for a window B_i is calculated by aggregating the target likelihood map $T(\mathbf{x}|B_i;\mathbf{c})$ scores of pixels inside the window region as

$$L(B_i;\mathbf{c}) = \alpha_i \sum_{\mathbf{x}\in B_i} T(\mathbf{x}|B_i;\mathbf{c}), \tag{2}$$

where $\mathbf{x} = [x, y]$ is a pixel inside the window B_i and α_i is an normalization factor that is inversely proportional to the window size.

In the target likelihood map $T(\mathbf{x}|B;\mathbf{c})$, the value of each pixel $\mathbf{x}$ indicates its likelihood belonging to the target object. It is accumulated over individual confidence maps of the windows that are generated by the set of object proposals in the current frame

$$T(\mathbf{x}|B;\mathbf{c}) = \lambda_s(\mathbf{x}) \sum_{i=1}^{N} p(\mathbf{x}|B_i)\lambda_c(\mathbf{x}|B_i;\mathbf{c}) \tag{3}$$

where $\lambda_c(\cdot)$ and $\lambda_s(\cdot)$ are the two terms that imposes proximity to initial click and similarity to appearance and motion based saliency, respectively. Notice that, the target likelihood map also uses the candidate proposals by their accumulating probabilities. In this mixture model, each proposal contributes to the likelihood map with its confidence scores corresponding to its model over pixels. As a result, the target likelihood map accumulates the confidences of all proposal windows.

For a given frame, the candidate windows B_i are generated by objectness methods. Each proposal is defined by its objectness score o_i, window center $\mathbf{x}_i$ and size (w_i, h_i). We model each window with a 2D Gaussian kernel as

$$p(\mathbf{x};B_i) = \frac{1}{2\pi|\Sigma_i|^{0.5}} \exp\left(-\frac{(\mathbf{x}-\mathbf{x}_i)^T\Sigma_i^{-1}(\mathbf{x}-\mathbf{x}_i)}{2}\right) \tag{4}$$

where Σ_i is the 2×2 covariance matrix computed for each window.

Using all proposals, we accumulate the $T(\mathbf{x}|B;\mathbf{c})$ by Eq. 3 and Eq. 4. We then compute the likelihood scores $L(B_i;\mathbf{c})$ for each proposal window B_i by Eq. 2.

A. Click Term λ_c

Object proposal algorithms find many object-like regions, including both target we want and other objects in background. To distinguish the user-specified target from the possible distractors, the user click plays a critical role.

As shown in Fig. 2, the human click distribution resembles a multivariate Gaussian function around the original object center. Based on this observation, the probability distribution of the click location can be modeled by a Gaussian kernel centered at the original window center $\mathbf{x}_{gt}$. Given a human click, the probability distribution of the target location can be modeled by a same distribution, which centered at $\mathbf{x}_c$ instead. Therefore, we define the click term each pixel with a 2D Gaussian distribution

$$\lambda_c(\mathbf{x}|B_i;\mathbf{c}) = \frac{1}{2\pi\sigma_m^2} \exp\left(-\frac{|\mathbf{x}-\mathbf{x}_c|^2}{2\sigma_m^2}\right) \tag{5}$$

where σ_m is positively proportional to the square root of the frame size m, and further weighted by a sigmoid function of the average motion magnitude term α_m of the frame as

$$\sigma_m \propto \frac{\sqrt{m}}{1+e^{-\alpha_m}} \tag{6}$$

where α_m is the average norm of the optical flow vectors between two successive frames.

We observe that the erratic motion sequences always contain large motion magnitudes and large center errors. As a result,

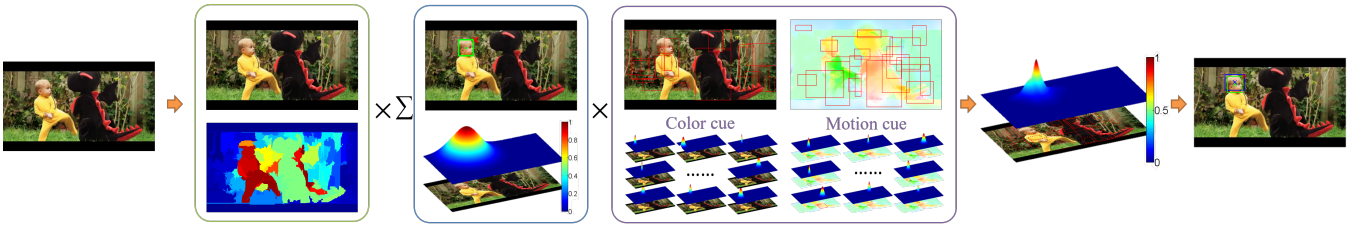


Fig. 5. The pipeline of the refinement method. (a) The input frame. (b) The computed saliency weight term. (c) The click model is a Gaussian kernel centered at the click position with a fixed covariance matrix learned from the dataset. The red cross denotes the user click and the green rectangle denotes the ground-truth region. (d) Candidate windows are generated by the objectness algorithm using both color and motion cues. (e) The TLM accumulated with individual confidence maps of all the windows weighted by the input click and the saliency. The window likelihoods are calculated. (f) The target is the maximum likelihood estimate. The blue window shows the refined window.

we extend the weighting term to a larger region around the human click position. However, the effectiveness of the magnitude weight may vary in different sequences. To address this, we select the proposals that contain the input human click within their windows. This leads to a more consistent target likelihood map $T(\mathbf{x}|B;\mathbf{c})$ that efficiently suppresses the background clutter.

B. Object Proposals B_i

We use both color and motion cues to generate object proposals. The object proposal algorithms [37], [38], [36], [39] are initially designed to work on color gradients.

For both color and motion cues, we select the same number of candidates with the largest objectness scores. Notice that, in some frames target movements are relatively small, consequently resulting in insufficient proposals from the motion cue, in which case we use the candidates only from the color cue.

1) MCG: Multiscale Combinatorial Grouping (MCG) [36] is a unified approach for bottom-up segmentation and object candidate generation. At its core are a fast eigenvector computation for normalized-cut segmentation and an efficient algorithm for combinatorial merging of hierarchical regions [40], like [41] and [42]. It improves on a multi-scale hierarchical segmentation [43].

2) BING: Instead of the segmentation and grouping strategy, Binarized Normed Gradients (BING) [38] uses a trained linear classifier upon edges and computes scores for candidate windows using efficient representations, which makes this pipeline very fast.

3) EdgeBoxes: [37] measures the difference the number of contours that are wholly contained in a bounding box and those crossing the box boundaries, and uses this difference as an indicator of the likelihood of the box containing an object. Using efficient data structures, it can evaluate millions of candidate boxes in a fraction of a second, returning a ranked set of a few thousand top-scoring proposals.

4) RPN: The *faster R-CNN* [39] introduces a Region Proposal Network (RPN) to generate candidate windows for the object detection task. It is embedded in the fast R-CNN [44] framework, with which shares the convolutional features. The reported detection precision using the RPN outperforms the *Selective Search* [41] and EdgeBoxes with less proposals.

We employ all these four proposal algorithms in our refinement framework using their pretrained models that are supplied by their authors to determine the most suitable object proposal method for automatic initialization of object tracking. Motion gradients are computed by optical flow where we use the fast version of the large distance optical flow [45].

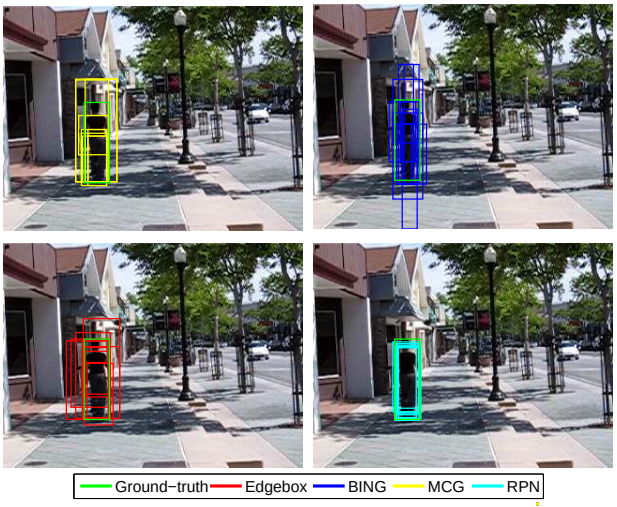


Fig. 6. 10 best-matched proposed windows for each object proposal algorithm.

Fig. 7. Sample saliency maps of initialization frames using DRFI [46].

C. Saliency Term λ_s

In addition to the click weight λ_c that suppresses the proposals further away from the click position, the target likelihood map is also weighted by a saliency confidence map

Algorithm 1: Refinement method

Proposals:

1. Generate candidate windows B_i by selecting a fixed number of the highest ranking proposals with respect to their objectness scores using both appearance and motion cues.

Target Likelihood Map:

2. Calculate the click term λ_c by Eq. 5.
3. Calculate the saliency term λ_s.
4. Accumulate the target likelihood map $T(\mathbf{x}|B;\mathbf{c})$ from all candidate windows by Eq. 3.
7. Calculate the window likelihood scores by Eq. 2.
8. Select the final window by Eq. 1.

λ_s to emphasize on the salient parts of the image. Here, we employ the Discriminative Regional Feature Integration (DRFI) [46], to generate saliency scores for all pixels. DRFI fuses a set of multilevel saliency maps $\{S_1, ..., S_L\}$ such that

$$\lambda_s(\mathbf{x}) = \sum_{l=1}^{L} w_l S_l(\mathbf{x}). \tag{7}$$

At each level, a pixel $\mathbf{x}$ is scored by a regional saliency descriptor integrating the local contrast, local appearance and local objectness. In comparison to other saliency methods, DRFI achieves superior performance [47]. For sample saliency results, see Fig. 7.

V. Experiment Results

We evaluate two aspects of the refinement strategy: the effectiveness of compensating for the click center error, and the ability to recover a target window size from a single click.

In addition, we analyze different combinations of objectness and saliency cues to discover an optimal solution. The best results are obtained using EdgeBoxes with color and motion cues with saliency term. MCG outperforms others on sequences with severe boundary blur. RPN achieves the leading performance for pedestrian targets. There are several factors affecting the performance of the refinement process and the details are discussed in the following sections.

In Figure 8, we show sample refinement outputs that illustrates the proposed refinement strategy is effective under various circumstances. Even for the clicks outside the target region, the refinement strategy recovers the correct target windows.

A. Experimental Settings

We use the real human inputs in HIC dataset collected on T-B50 as explained in Section III. Since the human clicks exhibit an acceptable accuracy for stationary frames, we focus on the erratic motion sequences where human clicks are frequently outside the ground-truth object region. We highlight the results for the erratic sequences and compare them against the human input performance. Similarly, since the *Pause&Click* strategy has limited versatility in practical applications, we concentrate on the *Follow&Click* and *Spot&Click* scenarios.

The HIC dataset contains about 18,980 clicks (11,207 for *Follow&Click* scenario and 7,773 for *Spot&Click* scenario), approximately 400 clicks from each sequence that cover the targets within an average span of 500 frames. These clicks can be entered at any arbitrary frame in a video to simulate real-life settings of tracker initialization. The reported results in this section for the erratic motion are based on 11 sequences and 4176 clicks where 2419 clicks are from *Follow&Click* and 1757 from *Spot&Click*.

To evaluate the refinement performance, we employ two measures: the center error, which measures the distance between the refined and ground-truth object window centers, and the overlap between the refined and the actual ground-truth windows, i.e. intersection-over-union (IoU). For objective and intuitive comparisons, the results are obtained for normalized the target windows on a 50×50 unit size in every frame.

B. Performance for Smooth and Erratic Motion

As discussed in Section III-D, the erratic motion sequences exhibit more challenging initialization task for human operators, thus a refinement process is essential. We assess the refinement performance with randomly selected clicks from the HIC dataset. The results of the erratic motion sequences are shown in Table III to VI.

Center Error: Figure 9 shows the center error graphs for 16 different component combinations for erratic motion sequences. The horizontal axis corresponds to increasing average velocities and the vertical axis is the average error.

As seen, the refinement method consistently and effectively reduces the center error of human clicks. The refinement method using EdgeBoxes proposals and color and motion saliency cues can reduce the center error more than 28.8% on average (from 35.33 pixels of original human click errors to 25.15 pixels) for the *Spot&Click* scenario and 26.9% on average (from 34.03 pixels of original human click errors to 24.89 pixels) for erratic motion sequences under the *Follow&Click* scenario.

For the sequences where the average velocity of the target is larger than 6 pixels per frame, human accuracy degrades significantly. Based on Table I, we see that the spatial error increases almost 85%, from 16.66 pixels on average for the smooth motion sequences, to 30.92 pixels for the erratic motion sequences. Especially for *Couple*, *Deer* and *Human9*, the center error goes over 40 pixels due to unpredictable target movements.

For the sequences where the targets are stationary or very slow with less than 6 pixels per frame average velocity (such as *Bolt* and *Redteam*), the improvement is limited due to the upper precision bound of the objectness proposal algorithm (explained in Section V-C and shown in Fig. 12).

Overlap Ratio: Reducing the center error is one aspect of the initialization accuracy. Since only a single click is provided, the size of the target has to be recovered as well. To assess the accuracy of the window size estimate, we employ the overlap ratio. We compare against two baseline strategies

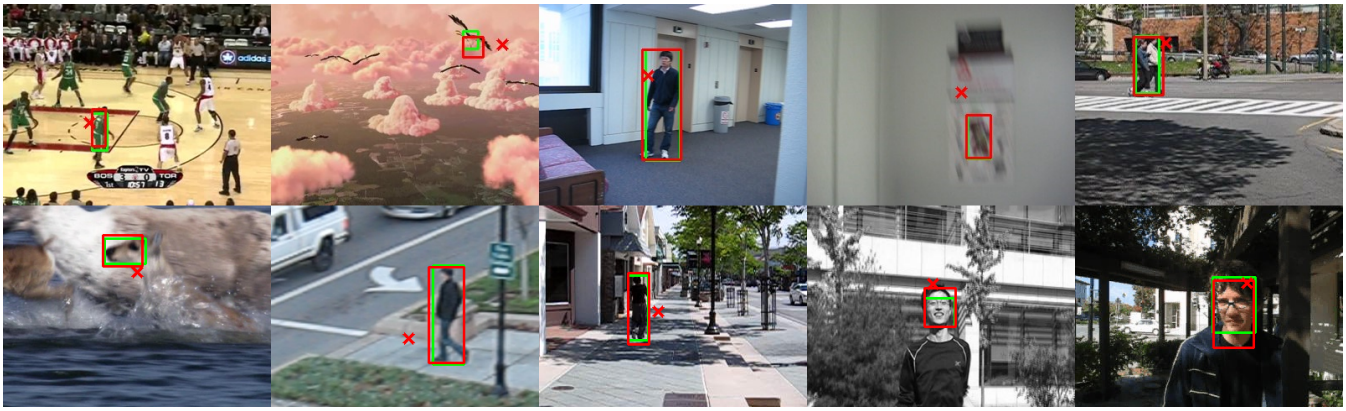


Fig. 8. Sample refinement results for inaccurate clicks; the proposed refinement method achieves to locate the correct windows. The human click positions are represented by red crosses while the ground-truth windows and the refined windows are represented by green and red windows, respectively.

that use object size statistics of the TB50 dataset: setting the object window by randomly sampling from all possible target sizes, and setting the object window as the average target size.

As shown in Table V and Table VI, and Fig. 10, the refinement method provides more accurate window sizes for all target velocities including the stationary targets. For the erratic motion sequences, the refinement strategy threefold increases the overlap ratio from 0.11 of original human clicks to 0.379 for the *Spot&Click* scenario, and from 0.137 to 0.376 for the *Follow&Click* scenario when the window size is set by random sampling. The improvement for the average window size assignment is also very significant; from 0.189 to 0.379 for the *Spot&Click* scenario, and from 0.202 to 0.376 for the *Follow&Click* scenario. Another observation is that while the human click is sensitive to the target velocity, the refinement method demonstrates a robust performance. Overall, the proposed method recovers more accurate window sizes.

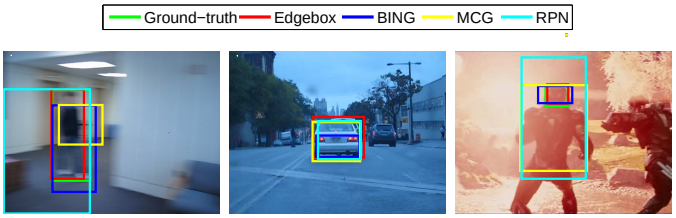
C. Choice of Objectness

We tested four state-of-the-art object proposal algorithms, BING [38], MCG [36], EdgeBoxes [37] and RPN [39] on the tracking dataset TB50. The recall responses are shown in Table VII for up to 3000 proposals. RPN uses the original *VGG-16* model provided by its authors.

TABLE VII
RECALL RATE OF OBJECT PROPOSAL ALGORITHMS ON TB50 INITIALIZATION FRAMES

BING[38]	EdgeBox[37]	MCG[36]	RPN[39]
96.53%	83.12%	81.55%	55.25%

In general, object proposal algorithms are evaluated on object detection benchmarks [48][49][50]. However, visual tracking task and its representative datasets have their own challenges. For instance, motion often leads into blurred boundaries, which may confuse the gradient-based object proposal techniques. Moreover, a tracked target can be only a partial object (such as head of a person rather than whole body). These differences are illustrated in Fig. 11. Our observation here is that the state-of-the-art object proposal methods have inherent limitations and they may not be able to select a perfect match even given a large number of proposals. As seen in Fig. 12, the object proposal methods can hardly find a perfect window considering both the center error and the overlap ratio. Similarly, we can see in Table V, Table VI, Fig. 9, and Fig. 10 that object proposal methods have different refinement results. Our refinement method improves upon this shortcoming of the object proposals by incorporating saliency and click terms, and reranking the proposals with respect to their likelihood scores.



(a) (b) (c)

Fig. 11. Characteristic challenges of object tracking videos for conventional object proposal methods. (a) Blurred boundaries. (b) Actual target is larger than object proposals. (c) Actual target is smaller than object proposals.

To better understand how object proposal methods perform for typical object tracking sequence, we measure the generated proposals' precision by an maximum overlap and a minimum center error scores. Since some candidates have small center error yet a low overlap ratio, which renders them unwanted, we calculate the minimum center error only within the set of highly-overlapped 10 candidates. Results are shown in Fig. 12. When the number of proposals increases over a certain number, the improvement becomes negligible. For computational consistency of the refinement method, we fix the number of maximum proposals to 3000 for all methods.

In recent surveys [48], MCG [36] outperforms the BING and EdgeBoxes methods. However, when evaluated by the center error and overlap, it fails to keep the leading place (see Fig. 12). One reason is that, MCG tends to find entire object regions rather than partial ones. We calculate the average proposal size from the three methods, MCG has a 228×166

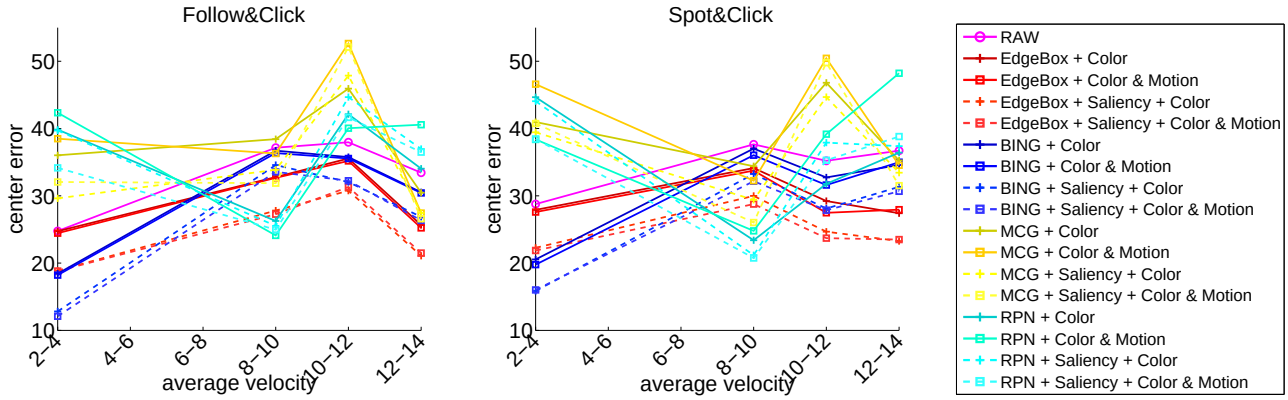


Fig. 9. Average center error graphs of different refinement methods for increasing target velocities for the erratic motion under the *Follow&Click* and *Spot&Click* scenarios (lower is more accurate).

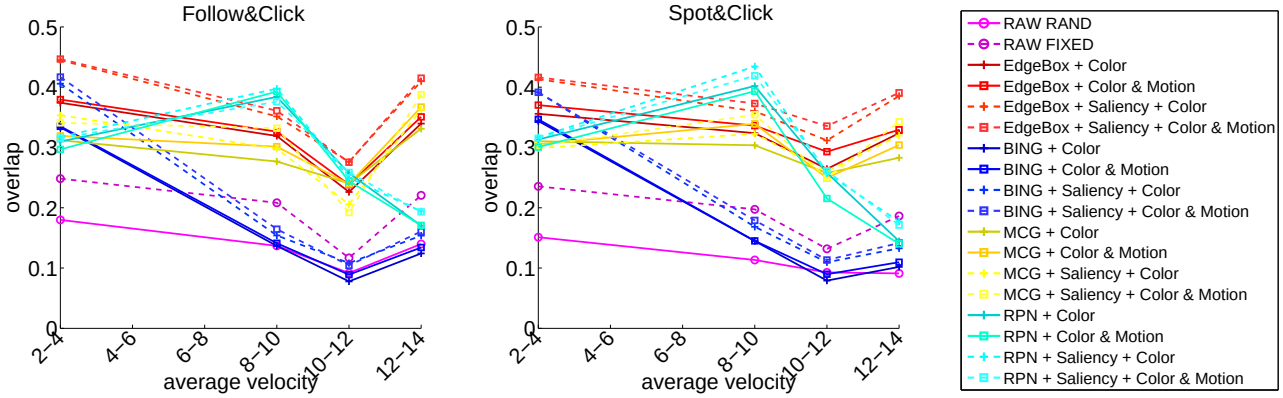


Fig. 10. Average overlap rates of different refinement methods for increasing target velocities for the erratic motion under *Follow&Click* and *Spot&Click* scenarios (higher is more accurate).

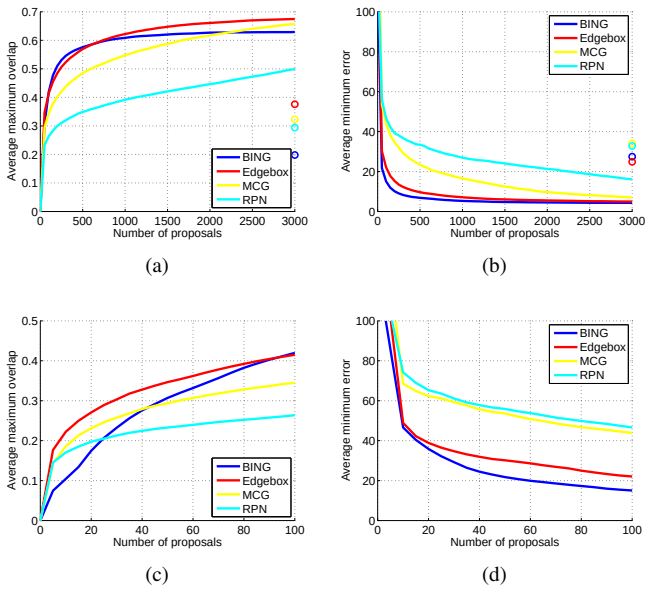


Fig. 12. The graphs show the accuracy of the most fitting (having the highest overlap) object proposals to the ground-truth object regions for the initialization frames of the TB50 dataset videos. The dots show the performance after the refinement process. (a) The maximum overlap. (b) The minimum center errors within 10 highly-overlapped candidates. (c-d) Graphs for up to 100 proposals.

average size while BING and EdgeBoxes are 50×61 and 75×71. Therefore MCG struggles to refine human inputs when the targets are small partial objects, e.g. *Deer*, *DragonBaby* and *surfer*. On the contrary, on the sequences with entire objects, like *BlurBody*, *BlurCar2*, *BlurFace*, *BlurOwl* and *Couple*, it defeats the other proposal methods.

RPN [39] has similar shortcomings when it comes to detection of small objects. However, it is highly effective for pedestrians. On sequences *BlurBody*, *Couple*, *Human9* and *Woman*, the average center error reaches 16.20 pixels and the overlap is 0.503, which is considerably better than the other proposal algorithms.

We observe the BING-based [38] schemes stay competitive in terms of the center error (see Table III and IV). In contrast to MCG, they perform inferior when evaluated by the overlap score (see Table V and VI.) due to the excessive number of relatively small candidates inside the true object regions (see Fig. 11(b)).

Considering all, EdgeBoxes [37] remains as the best solution not only for the erratic motion sequences but also for the whole dataset. It achieves robust detection for tracking of arbitrary targets. In case of prior information, such as object type, becomes available, it is possible to choose a more suitable proposal generation method. It could also be an interesting experiment to combine different proposal methods.

TABLE III
CENTER ERROR IN ERRATIC MOTION, *Follow&Click* SCENARIO

SEQUENCE	RAW	RPN		MCG		BING		EdgeBox	
		COLOR	C&M	COLOR	C&M	COLOR	C&M	COLOR	C&M
BlurBody	30.22	11.41	15.91	18.09	16.08	30.99	30.64	27.94	26.91
BlurCar2	19.69	11.16	14.03	13.44	11.45	20.21	20.03	11.77	11.53
BlurFace	29.90	30.84	30.98	13.54	13.96	27.30	27.89	21.35	21.46
BlurOwl	35.64	27.47	35.20	13.99	12.85	20.19	17.74	10.21	10.37
Couple	43.49	19.47	20.10	22.03	25.30	35.20	35.20	34.62	33.13
Deer	45.68	77.92	67.39	77.64	88.12	33.08	33.84	33.58	35.72
DragonBaby	34.77	52.56	43.29	54.83	52.98	33.10	33.41	31.70	32.65
Human9	47.80	25.14	19.61	49.41	49.75	46.25	44.33	44.89	45.66
Jumping	37.61	46.37	46.12	50.59	41.02	36.65	34.69	19.93	18.72
Surfer	31.32	71.51	60.90	39.95	43.74	16.20	15.39	22.94	22.94
Woman	18.21	7.89	7.41	19.28	20.37	9.42	8.79	14.83	14.69
Average	34.03	34.70	32.81	33.89	34.15	28.05	27.45	24.89	**24.89**

TABLE IV
CENTER ERROR FOR ERRATIC MOTION, *Spot&Click* SCENARIO

SEQUENCE	RAW	RPN		MCG		BING		EdgeBox	
		COLOR	C&M	COLOR	C&M	COLOR	C&M	COLOR	C&M
BlurBody	29.74	10.33	17.25	13.35	12.75	25.82	25.40	24.20	21.79
BlurCar2	19.49	11.83	11.63	10.05	9.58	18.76	18.74	9.68	9.62
BlurFace	36.45	33.32	33.74	14.29	13.47	32.29	32.30	23.69	23.58
BlurOwl	40.40	34.13	41.12	25.68	22.26	30.62	29.61	18.63	18.26
Couple	47.89	21.28	19.60	30.55	26.46	42.08	39.11	43.31	44.32
Deer	40.67	65.47	53.32	76.01	86.83	30.38	30.31	25.12	25.61
DragonBaby	33.28	44.71	41.57	60.35	58.80	31.08	30.10	27.52	28.64
Human9	38.96	14.17	13.49	42.66	42.60	33.73	34.03	48.05	45.77
Jumping	44.24	37.41	38.28	34.32	25.46	38.42	36.76	18.88	15.37
Surfer	33.06	80.79	68.27	52.83	55.45	17.73	18.10	26.01	26.01
Woman	24.49	7.33	8.62	26.04	26.04	13.97	13.91	18.53	17.73
Average	35.33	32.80	31.53	35.10	34.52	28.63	28.04	25.78	**25.15**

We show the accuracy of the refinement process[1] in Fig. 12(a) and Fig. 12(b), denoted by colored dots. In general, the refined outputs are not necessarily the highest-overlapped object proposals. Since the refinement process only utilizes unreliable human clicks and low-level image cues, it is unfair to compare the refinement process results with the best possible object proposals. To our advantage, the shown performance gap indicates our proposed method can be improved further when more competent object proposal methods become available.

D. Influence of Motion Cue:

We extend the appearance based object proposals to motion, in particular using short-term motion gradient information, in our refinement stage. For the sequences with large object-camera motion, the motion cue is indispensable. As expected, for the sequences with slow or stationary targets, the motion cue may not be reliable enough to generate accurate proposals. For RPN, the motion cue may even slightly degrade the accuracy. Another observation is that human operators tend to click at the frames with slow or stationary targets where short-term optical flow motion boundaries may be insignificant. Therefore, the motion cue can be considered unnecessary for refinement for slow or stationary targets.

[1]We employ 3000 proposals in the refinement process.

E. Choice of Saliency Cue

We evaluate the influence of the saliency cue by running versions with and without saliency. We show the influence of saliency cue on erratic motion sequences in Table VIII using the clicks from *Follow&Click* scenario (clicks from *Spot&Click* scenario have similar responses). The most improved results are achieved when the target likelihood map is weighted by the saliency terms, which suppress the irrelevant candidates and clutter in the background. For RPN, the center error becomes worse for the sequences (*BlurFace*, *Deer*, *DragonBaby* and *Jumping*) with small and partial objects.

TABLE VIII
INFLUENCE OF SALIENCY. CENTER ERROR [OVERLAP RATIO]

	RPN	MCG	BING	Edgebox
No Saliency	34.81[0.288]	37.24[0.311]	31.29[0.165]	29.60[0.327]
Saliency	32.81[0.294]	34.15[0.323]	27.45[0.198]	**24.89[0.376]**

F. Influence on Trackers

We evaluate the influence of the refinement process on different trackers. Each tracker is first initialized with 100 original human clicks using random object size, and then for comparison, initialized with the corresponding results of the refinement process. Similar to [11], the tracking performance is reported in two aspects: *success plot* and *precision plot*.

TABLE V
OVERLAP FOR ERRATIC MOTION, *Follow&Click* SCENARIO

SEQUENCE	RAW		RPN		MCG		BING		EdgeBox	
	RAND	FIXED	COLOR	C&M	COLOR	C&M	COLOR	C&M	COLOR	C&M
BlurBody	0.087	0.100	0.438	0.425	0.322	0.335	0.044	0.045	0.227	0.223
BlurCar2	0.155	0.260	0.499	0.459	0.490	0.520	0.079	0.080	0.501	0.506
BlurFace	0.120	0.189	0.207	0.211	0.425	0.414	0.084	0.083	0.254	0.254
BlurOwl	0.137	0.230	0.197	0.183	0.431	0.546	0.224	0.247	0.651	0.651
Couple	0.130	0.216	0.508	0.477	0.350	0.336	0.189	0.190	0.275	0.290
Deer	0.096	0.133	0.061	0.093	0.090	0.050	0.172	0.164	0.325	0.328
DragonBaby	0.163	0.242	0.179	0.185	0.234	0.204	0.153	0.152	0.325	0.340
Human9	0.106	0.176	0.411	0.443	0.139	0.138	0.148	0.157	0.208	0.215
Jumping	0.156	0.181	0.173	0.125	0.214	0.333	0.200	0.231	0.421	0.432
Surfer	0.135	0.167	0.121	0.106	0.306	0.281	0.498	0.507	0.487	0.487
Woman	0.224	0.330	0.514	0.525	0.400	0.400	0.313	0.327	0.403	0.407
Average	0.137	0.202	0.301	0.294	0.309	0.323	0.191	0.198	0.371	**0.376**

TABLE VI
OVERLAP FOR ERRATIC MOTION, *Spot&Click* SCENARIO

SEQUENCE	RAW		RPN		MCG		BING		EdgeBox	
	RAND	FIXED	COLOR	C&M	COLOR	C&M	COLOR	C&M	COLOR	C&M
BlurBody	0.099	0.093	0.459	0.407	0.436	0.448	0.047	0.049	0.249	0.284
BlurCar2	0.141	0.251	0.508	0.498	0.550	0.566	0.088	0.092	0.559	0.563
BlurFace	0.063	0.153	0.216	0.228	0.418	0.433	0.053	0.062	0.262	0.265
BlurOwl	0.089	0.185	0.145	0.120	0.393	0.452	0.207	0.221	0.582	0.590
Couple	0.072	0.180	0.538	0.524	0.303	0.307	0.153	0.174	0.223	0.217
Deer	0.086	0.171	0.056	0.115	0.084	0.051	0.172	0.178	0.374	0.387
DragonBaby	0.121	0.220	0.171	0.166	0.149	0.143	0.139	0.141	0.312	0.317
Human9	0.127	0.195	0.484	0.512	0.158	0.160	0.215	0.212	0.212	0.239
Jumping	0.114	0.164	0.208	0.142	0.280	0.384	0.218	0.238	0.447	0.473
Surfer	0.102	0.170	0.108	0.096	0.226	0.238	0.492	0.494	0.442	0.442
Woman	0.200	0.301	0.526	0.534	0.372	0.367	0.291	0.289	0.385	0.391
Average	0.110	0.189	0.311	0.304	0.306	0.323	0.188	0.195	0.368	**0.379**

These plots show the success rate and precision for varying overlap and center error thresholds, respectively. In addition, the area under curve (AUC) is used to score each tracker (shown in parentheses).

As in Fig. 13, unreliable human click initializations lead to severely degrared performance in comparison with the experiments where the tracker is initialized by the ground-truth target regions [11]. Among the tested trackers, Struck [2], TLD [5] and CSK [51] are observed to be less sensitive to initialization errors than IVT [52], CT [53], DFT [17], and ORIA [54]. For all tested trackers, our refinement process clearly improves the performance. The average AUC score increases by 0.088. We also notice that, in the success plot, the AUC scores increase 0.159, 0.139, 0.127 for Struck, TLD, and CSK, while the improvement is 0.065, 0.054, 0.052, 0.020 for CT, IVT, DFT, and ORIA. This indicates that the accuracy improvement is more advantageous for less sensitive trackers.

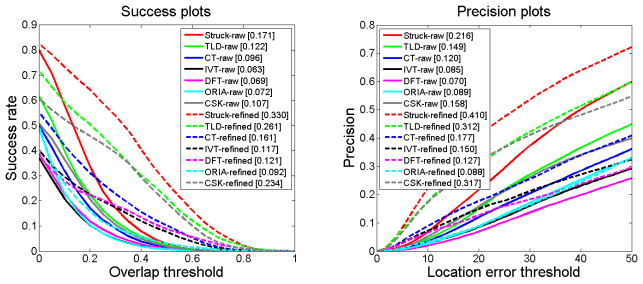


Fig. 13. Influence of the refinement process on tracking performance. Each tracker is initialized with 100 regions centered on the original human clicks and object size is randomly assigned from the distribution of the ground-truth object sizes (solid graphs), and then with 100 regions obtained after the refinement process using the corresponding human clicks (dashed graphs). As visible, the refinement consistently improves the performance for all trackers. Best viewed on screen and in color.

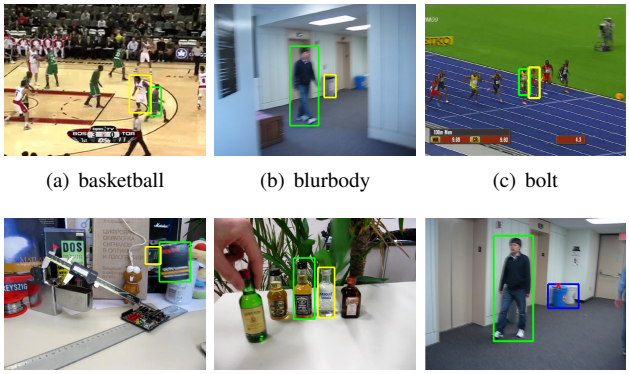
G. Performance with Multiple Objects

In Section III, we have the observation that human clicks follow a multivariate Gaussian distribution around the original object center. In most frames, human clicks are located near to correct targets. However, erroneous clicks still exist due to the relative object-camera motion.

In a set of additional experiments, we test our method's performance under multiple objects. To this end, we label the closest objects to the target as *distractors*. As shown in Figure 14(a) to 14(e), distractors may be located close to the actual targets, a case that may easily cause human operator to click on the wrong objects. Denoting O_{wrong} as the overlap between distractors and our refinement outputs, and O_{right} as the overlap between the actual targets and our refinement outputs, an initialization is classified as a miss (incorrect) if $O_{wrong} > O_{right}$. From the TB50 dataset, we select five sequences in which distractors always move around and close to the actual targets.

As reported in Table IX, the miss ratios after the refinement are 11%, 24%, 31%, 1% and 1% for the selected sequences,

respectively. These results indicate that, for most clicks, our proposed method finds the correct objects. There are a few failure cases we observed. Figure 14(f) shows a failure case in which the operator clicks directly on the wrong object. We note that our method is agnostic to object class (i.e. it is not aiming at detecting humans or vehicles only) and designed to work for a general class of objects. Without any prior information, even a human operator cannot tell which object would be the intended target by just based on a given click location.



(a) basketball (b) blurbody (c) bolt

(d) box (e) liquor (f) a miss output

Fig. 14. (a)-(e) Targets (green) and distractors (yellow). (f) Failure case: an initial click (red) and its corresponding initialization after the refinement process (blue).

TABLE IX
MISS RATIO OF THE REFINEMENT PROCESS FOR MULTIPLE OBJECTS

sequence	basketball	blurbody	bolt	box	liquor
miss ratio	0.11	0.24	0.31	0.01	0.01

H. Computational Load

We analyze the number of operations and also report the run-time for each component of our refinement method.

Denoting N as the number of proposals, m is the size of image, the saliency computation DRFI [46] attains a linear time complexity $O(m)$ in terms of the image size. The most computationally intensive processes are the initial segmentation, which is obtained by a graph-based segmentation approach [55], and the computation of region descriptors, which are encoded by a 86-dimensional vector. Both of these processes have linear time complexity to the input image size m and can be implemented efficiently in a parallel processing architecture.

In terms of the object proposal stage, MCG implementation[2] is the least efficient solution. In CPU versions, MCG takes more than 10 seconds for a frame with VGA resolution, while BING, Edgeboxes and RPN have linear complexity $O(m)$. In RPN, the forward-propagation (feature computation) of the CNN has complexity $O(mf^2lc)$, where f is the number of feature maps in one layer, l is the number of layers, and c is the time complexity of one convolution operator. We note that the specific CNN version we used is designed for GPU architectures and runs in real-time. (BING can achieve above 300 fps, RPN 100 fps, and EdgeBoxes 30 fps as reported in the past.)

Complexity of the target likelihood map computation stage is $O(Nm)$ since we calculate the confidence value for each pixel using Eq. 4. The pixel-wise scores can be computed in parallel, thus the time complexity can be optimized to $O(m)$. In short, the component solutions we used in our framework can be implemented on a parallel processing architecture for real-time performance.

The run-times of our unoptimized implementation can be found in Table X. Hardware platform we used is Xeon E5-2650 @ 2.0 GHz CPU with 16G RAM. Average frame resolution is 526 $\times$ 354 pixels. The target likelihood map, proposal generation, and optical flow can be calculated within about 0.6 seconds in all. The bottleneck is the saliency computation, i.e. DRFI, which takes 1.8 seconds for one frame.

The proposed method has a high potential to achieve real-time performance based on two considerations: Firstly, it can be accelerated with more efficient implementations. Both TLM and DRFI computations are parallelizable, which can be accelerated remarkably on parallel processing hardware (e.g. on a GPU). DRFI reportedly achieved 0.418s with a similar image resolution [46]. Besides, the TLM and edgeboxes implementations are programmed with Matlab code which can be accelerated by using another language, like C/C++.

Secondly, our refinement method is flexible in selecting various components as discussed in Section V-C and V-E. This means, it is possible to choose more efficient proposal, saliency, and optical flow generation methods such as BING that runs at 300 fps, minimum barrier saliency [56] at 80 fps. As demonstrated in Section V-E, our method can even perform competently without the saliency cue.

TABLE X
RUN-TIMES OF THE PROPOSED REFINEMENT FRAMEWORK

Process	Run-time	Code
Refinement Process	0.1555	Matlab
Proposal (Edgeboxes [37])	0.2908	Matlab
Optical Flow (EPPM [57])	0.1507	C
Saliency (DRFI [46])	1.802	C
Overall	2.399	

VI. CONCLUSION

In this paper, we provided a systematic analysis of human initialization accuracy. We introduced a new human initialization click (HIC) datasets that contains more than 20,000 human initializations. We discussed three operation scenarios and underlined the influences of target velocity and target size on the accuracy of human clicks.

In addition, we presented an automatic refinement strategy that can leverage any state-of-the-art trackers into human-centric interactive tracking solutions. We analyzed multiple object proposal and image saliency methods to determine the optimal choice of components. Our extensive evaluations show that the initialization errors can be reduced up to 30% and the overlap ratio can be increased 3$\times$.

[2] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/mcg/

Since target initialization is an integral yet under-analyzed component of object tracking, this work naturally bridges the gap between the state-of-the-art object tracking algorithms and realistic human inputs. It offers a practical solution for real-world applications.

REFERENCES

[1] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 3, pp. 583–596, 2015. 1

[2] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 263–270. 1, 2, 12

[3] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2. IEEE, 2000, pp. 142–149. 1

[4] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 983–990. 1

[5] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 7, pp. 1409–1422, 2012. 1, 2, 12

[6] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1245–1256, 2013. 1, 2

[7] K. M. Yi, H. Jeong, B. Heo, H. J. Chang, and J. Y. Choi, "Initialization-insensitive visual tracking through voting with salient local features," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2912–2919. 1, 2

[8] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014. 1

[9] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Computer Vision-ECCV 2014 Workshops*. Springer, 2014, pp. 254–265. 1

[10] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: an experimental survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 7, pp. 1442–1468, 2014. 1, 2, 4

[11] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Computer vision and pattern recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2411–2418. 1, 2, 3, 4, 11, 12

[12] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojíř, G. Fernandez, A. Lukežič, A. Dimitriev *et al.*, "The visual object tracking vot2014 challenge results," in *Computer Vision-ECCV 2014 Workshops*. Springer, 2014, pp. 191–217. 1, 2, 4

[13] Y. Wu, J. Lim, and M. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 1

[14] A. Li, M. Lin, Y. Wu, M.-H. Yang, and S. Yan, "Nus-pro: A new visual tracking challenge," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 335–349, 2016. 2

[15] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: algorithms and benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5630–5644, 2015. 2

[16] T. B. Dinh, N. Vo, and G. Medioni, "Context tracker: Exploring supporters and distracters in unconstrained environments," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1177–1184. 2

[17] L. Sevilla-Lara and E. Learned-Miller, "Distribution fields for tracking," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1910–1917. 2, 12

[18] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern recognition*, vol. 13, no. 2, pp. 111–122, 1981. 2

[19] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004. 2

[20] J. Kwon and K. M. Lee, "Tracking by sampling trackers," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1195–1202. 2

[21] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Computer VisionECCV 2000*. Springer, 2000, pp. 751–767. 2

[22] K. K. Ng and E. J. Delp, "Object tracking initialization using automatic moving object detection," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2010, pp. 75 430M–75 430M. 2

[23] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 282–295. 2

[24] J. Lezama, K. Alahari, J. Sivic, and I. Laptev, "Track to the future: Spatio-temporal video segmentation with long-range motion cues," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011. 2

[25] Z. Chaohui, D. Xiaohui, X. Shuoyu, S. Zheng, and L. Min, "An improved moving object detection algorithm based on frame difference and edge detection," in *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on*. IEEE, 2007, pp. 519–523. 2

[26] M. Yokoyama and T. Poggio, "A contour-based moving object detection and tracking," in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. IEEE, 2005, pp. 271–276. 2

[27] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 6, pp. 1187–1200, 2014. 2

[28] K. Fragkiadaki, G. Zhang, and J. Shi, "Video segmentation by tracing discontinuities in a trajectory embedding," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1846–1853. 2

[29] A. Ottlik and H.-H. Nagel, "Initialization of model-based vehicle tracking in video sequences of inner-city intersections," *International Journal of Computer Vision*, vol. 80, no. 2, pp. 211–225, 2008. 2

[30] S. Sivaraman and M. M. Trivedi, "Integrated lane and vehicle detection, localization, and tracking: A synergistic approach," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 14, no. 2, pp. 906–917, 2013. 2

[31] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1815–1821. 2

[32] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid, "Joint tracking and segmentation of multiple targets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5397–5406. 2

[33] V. Mahadevan and N. Vasconcelos, "Saliency-based discriminant tracking," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1007–1013. 2

[34] ——, "Automatic initialization and tracking using attentional mechanisms," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*. IEEE, 2011, pp. 15–20. 2

[35] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A survey," *arXiv preprint arXiv:1411.5878*, 2014. 2

[36] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 328–335. 6, 7, 9

[37] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 391–405. 6, 7, 9, 10, 13

[38] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "Bing: Binarized normed gradients for objectness estimation at 300fps," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 3286–3293. 6, 7, 9, 10

[39] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99. 6, 7, 9, 10

[40] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" 2015. 7

[41] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1879–1886. 7

[42] P. Krähenbühl and V. Koltun, "Geodesic object proposals," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 725–739. 7

[43] P. Dollár and C. L. Zitnick, "Fast edge detection using structured forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 8, pp. 1558–1570, 2015. 7

[44] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448. 7

[45] L. Bao, Q. Yang, and H. Jin, "Fast edge-preserving patchmatch for large displacement optical flow," *Image Processing, IEEE Transactions on*, vol. 23, no. 12, pp. 4996–5006, 2014. 7

[46] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, "Salient object detection: A discriminative regional feature integration approach," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2083–2090. 7, 8, 13

[47] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *Image Processing, IEEE Transactions on*, vol. 24, no. 12, pp. 5706–5722, 2015. 8

[48] J. Hosang, R. Benenson, and B. Schiele, "How good are detection proposals, really?" *arXiv preprint arXiv:1406.6962*, 2014. 9

[49] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2014. 9

[50] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255. 9

[51] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European conference on computer vision*. Springer, 2012, pp. 702–715. 12

[52] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008. 12

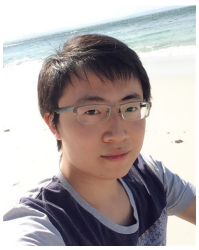
[53] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *European Conference on Computer Vision*. Springer, 2012, pp. 864–877. 12

[54] Y. Wu, B. Shen, and H. Ling, "Online robust image alignment via iterative convex optimization," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1808–1814. 12

[55] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004. 13

[56] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech, "Minimum barrier salient object detection at 80 fps," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1404–1412. 13

[57] L. Bao, Q. Yang, and H. Jin, "Fast edge-preserving patchmatch for large displacement optical flow," *Image Processing, IEEE Transactions on*, vol. 23, no. 12, pp. 4996–5006, 2014. 13



Hao Zhu is a PhD candidate at the State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing Institue of Technology. He received his BS degree in the School of Automation from Nanjing University of Science and Technology in 2011. He was a visiting student in the Research School of Engineering, Australian National University funded by a scholarship from the China Scholarship Council from September 2014 to September 2015. His current research interests include visual tracking and video segmentation.

Fatih Porikli is an IEEE Fellow and a Professor in the Research School of Engineering, Australian National University (ANU). He is also managing the Computer Vision Research Group at Data61. He has received his PhD from New York University in 2002. Previously he served as the Distinguished Research Scientist at Mitsubishi Electric Research Laboratories. His research interests include computer vision, pattern recognition, manifold learning, image enhancement, robust and sparse optimization and online learning with commercial applications in video surveillance, car navigation, robotics, satellite, and medical systems. He is the recipient of the R&D 100 Scientist of the Year Award in 2006. He won 5 Best Paper Awards at premier IEEE conferences and received 5 other professional prizes. He authored more than 150 publications, coedited two books, and invented 66 patents. He is serving as the Associate Editor of several journals for the past 12 years.