IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 11, 2020, accepted August 23, 2020, date of publication August 31, 2020, date of current version September 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3020309

POP: A Generic Framework for Real-Time Pose Estimation of Planar Objects

SEUNGHO CHAE[1], JE HYEONG HONG[1], HEESEUNG CHOI[1], AND IG-JAE KIM[1,2], (Member, IEEE)

[1]Imaging Media Research, Korea Institute of Science and Technology (KIST), Seoul 02792, South Korea
[2]Division of Nano and Information Technology, KIST School, University of Science and Technology (UST), Seoul 02792, South Korea

Corresponding author: Ig-Jae Kim (drjay@kist.re.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Project under Grant 2018M 3E3A1057288, and in part by the Korea Institute of Science and Technology (KIST) Flagship Project under Project 2E30270.

ABSTRACT Accurate pose estimation of planar objects is a key computation in visual localization tasks, with recent studies showing remarkable progress on a handful of baseline datasets. Nonetheless, achieving similar performance on sequences in unconstrained environments is still an ongoing quest to be accomplished, largely due to the existence of several sources of errors, which are correlated but often only partly tackled in the literature. In this article, we propose POP, a generic real-time planar-object pose-estimation framework which is designed to handle the aforementioned types of errors while not losing generality to a specific choice of keypoint detection or tracking algorithm. The essence of POP lies in activating keypoint detection module in the background as well as adding several refinement steps in order to reduce correlated sources of errors within the pipeline. We provide extensive experimental evaluations against state-of-the-art planar object tracking algorithms on baseline and more challenging datasets, empirically demonstrating the effectiveness of the POP framework for scenes with large environmental variations.

INDEX TERMS Planar object tracking, pose estimation, keypoint matching, structured output SVMs.

I. INTRODUCTION

Object pose estimation is central to many applications in computer vision and robotics, namely surveillance, robot manipulation and augmented reality (AR) [11], [44]. In particular, the problem of tracking pose variation of a planar object is regaining attention in AR for user localization as planar homographies provide strong point-to-point geometric constraints for relative pose estimation [16]. This is a challenging problem since users can move fast and rotate sporadically in an unpredictable way while enjoying their AR contents in the wild, inducing significant motion blurs, viewpoint shifts and illumination changes.

Estimating pose of a planar object has traditionally been carried out using a handcrafted modular pipeline, whereby features are initially extracted and matched, and the matched features are tracked using an optical flow algorithm (such as the Kanade-Lucas-Tomasi (KLT) tracker [32]). The tracked features are then used to estimate the underlying homography in each frame. Major baseline and state-of-the-art methods adopt this framework albeit accompanied by various modifications, with recent work [25], [44] achieving near 80–99% average tracking accuracies on several public datasets. Unfortunately, such positive results do not always replicate to unconstrained scenes encompassing large environmental variations [44], leaving it as one of the remaining challenges for this type of method. This is the main motivation of our work.

In recent years, several work have devised end-to-end frameworks for estimating pose of a planar object from raw images, largely motivated by the success of deep learning-based methods in estimating poses of non-trivial rigid objects [19]. Nevertheless, state-of-the-art results for planar objects are still achieved by more traditional modular pipelines [44] primarily due to lack of training data available for end-to-end platforms, which require large variations in backgrounds and viewpoints to learn the correct representation for planar objects [38]. This is the major bottleneck behind the current deep learning-based methods, especially in AR where scenes can exhibit large environmental changes.

The associate editor coordinating the review of this manuscript and approving it for publication was Shiping Wen.

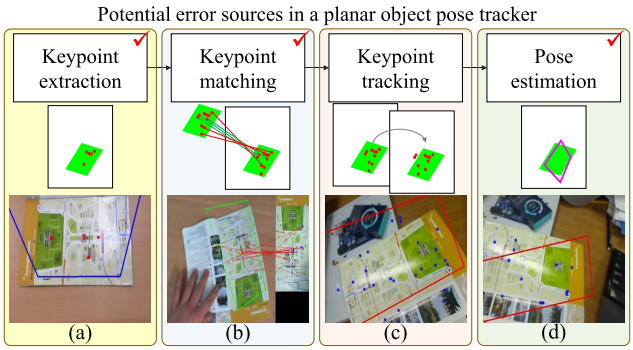


FIGURE 1. Illustration of potential error sources in estimating pose of a planar object through a traditional modularized pipeline. In (a), extracted keypoints are concentrated in a specific region or have low feature scores; (b) incorrect keypoint correspondences are formed between the initial image and the current image; (c) keypoint tracking fails (e.g. due to fast motion); (d) Homography is incorrectly estimated (e.g. due to outliers). We address the problem of reducing these error sources altogether.

While it is hoped that more training data would eventually yield end-to-end methods with desirable accuracy, we believe enhancing traditional modular pipeline to work better in unconstrained environments (frequently encountered in AR) would yield its own valuable contribution during this period of potential transition.

Modular pose estimation pipelines usually suffer from errors arising from multiple sources, namely low-quality or biased feature extraction, feature mismatches, unreliable keypoint tracking and incorrect homography estimation (see Fig. 1) for an illustration). Regrettably, these errors are not just additive but correlated, i.e. features concentrated in a specific region can trigger biased and potentially incorrect feature matches, increasing the possibility of incorrect homography estimation. While methods in the literature typically boils down to minimizing just one source of errors, the highly correlated nature of these errors inspires us to develop a framework jointly considering all the error sources. This leads to the following contributions:

+ We activate a keypoint detection module in the background to prepare for potential keypoint tracking failures, we manipulate [15]'s SSVM approach and implement PROSAC to utilize weights learned from SSVM in effectively ordering samples of correspondences in estimating homography.
+ In Sec. III-A, we propose a grid-level dynamic thresholding technique for retrieving useful keypoints relatively evenly across the query image in each frame.
+ Using the above as well as other refinement and learning modules, we propose a real-time planar object pose-estimation framework (POP, see Sec. III), which is capable of reducing bias in keypoint extraction, decreasing errors arising from feature mismatches and removing outliers for more stable homograph estimation (see Fig. 2).
+ We provide a thorough experimental evaluation of our POP framework against baseline and state-of-the-art planar object tracking methods across multiple challenging benchmark datasets (namely UCSB [13], Lintrack [50], EOS [15], MTSC [49], TMT [41] and POT210 [30]), empirically demonstrating the effectiveness of our scheme especially for unconstrained scenes.
+ We present our own unconstrained dataset (MSL), which has emphasis on large variations in viewpoints and zoom scales. Our POP scheme is again compared against other planar object pose estimation algorithms on this dataset.

Conversely, this work is limited to using off-the-shelf tracking modules, and consequently errors generated within the tracking process are not refined. Nevertheless, we believe this is a pioneering work for considering multiple sources of errors in a single pipeline, and it is hoped that this will provide a useful direction for future research in reducing correlated errors for planar object tracking.

II. RELATED WORK

Early learning techniques were performed using a predefined classifier offline. These methods are inevitably classifier because it does not adaptively respond to real-time environmental changes [44]. In recent years, an online learning methods have been actively studied in real-time input images [36]. In this section, we then summarize the main technique of the recent methods which are based on online learning for object tracking and pose estimation. In this section, we review relevant literature in pose estimation of planar objects.

A. PLANAR OBJECT POSE TRACKING

In the early days, a planar object's pose was estimated by attaching some markers around the object [23]. Marker-based trackers typically involve a template-based method for recognition and tracking [4]. The representative keypoint detector for marker detection is accelerated segment test (FAST) [40] and generic accelerated segment test (AGAST). These detectors are used to extract the corner points of each marker with which the tracker estimates the respective plane geometry. This type of trackers are computationally cheap but there is a big disadvantage that they can only be used in constrained environments where markers are present.

To address the above problem, studies began to focus on markerless tracking approaches (also known as natural feature tracking (NFT)), which can recognize objects by extracting features from images [37]. For finding feature correspondences, feature detectors such as FAST and AGAST are combined with a feature descriptor which characterizes local information around each keypoint. Such descriptors include binary robust independent elementary features (BRIEF) [8], binary robust invariant scalable keypoints (BRISK) [29], speeded-up robust features (SURF) [3]. Nowadays, most planar object tracking pipelines [25], [44] utilize NFT and optical flow to match or track features from which pose is estimated.

Recent work focused mostly on improving robustness to feature mismatches. [15] advanced the Struck [14] method (mentioned below) to specialize for planar objects using online structured support vector machine (SSVM), which updates pose of the target object in each frame. [49] considered the temporal spatial consistency of extracted keypoints

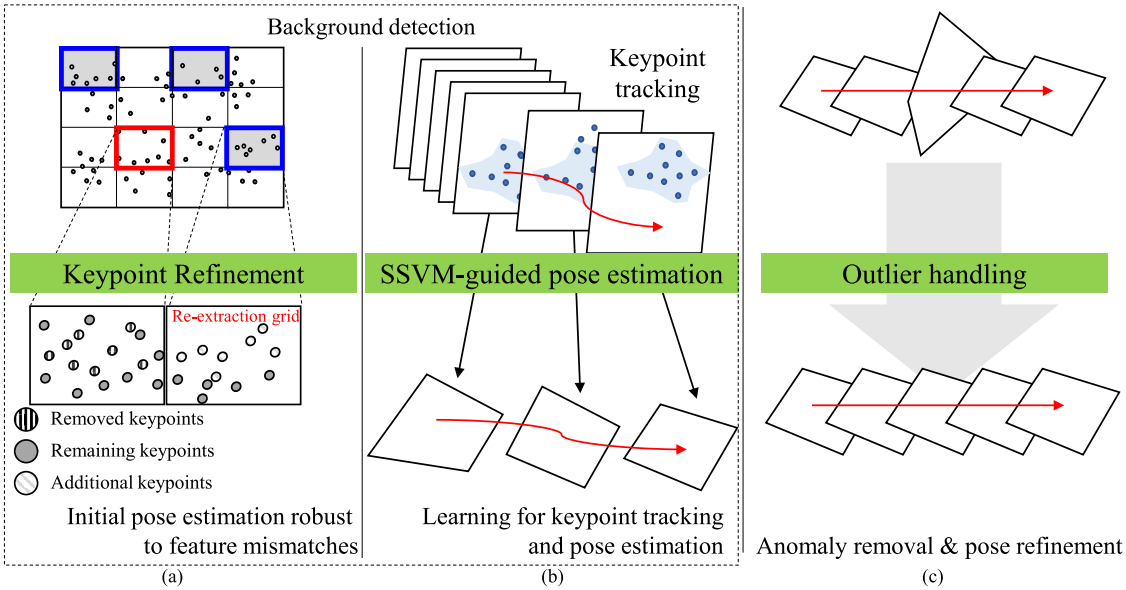


FIGURE 2. Main contributions of our POP framework. In (a), our keypoint refinement module tries to retrieve keypoints evenly across the image by dividing image into grids and dynamically changing the detection threshold in each grid. In (b), we apply online structured support vector machine to weigh correspondences (between the query and initial database images) through which homography is estimated. In (c), outlier correspondences are removed through inverse homography computation after which the homography is re-estimated.

TABLE 1. A summary of related work.

	Feature	Model estimation and update learning technique
[31]	Patch	Principal Component Analysis (PCA) using Sequential Karhunen-Loeve, Root Mean Square
[39]	Patch	Particle Filter, PCA, Hidden Markov Model
[26]	Patch	Basin Hopping Monte Carlo
[20]–[22]	Patch	AdaBoost, Normalized Cross Correlation, Lucas-Kanade
[2]	Patch	Online Boosting, Multiple Instance Learning
[27]	Patch	Bayesian Tracking, Spare PCA, Interactive Markov Chain Monte Carlo
[28]	Patch	Bayesian Tracking, Markov Chain Monte Carlo
[14]	Keypoint	Structured Support Vector Machine (SSVM)
[15]	Keypoint	Classification-based SSVM
[48]	Patch	Classification Score Random Projection
[5]	Keypoint	Structure-Aware Keypoint
[6], [7]	Patch	Graph matching, Spatial Prior with Graph cut
[25]	Patch	Particle filter, PCA, Normalized cross correlation
[49]	Keypoint	Match Score-based Structured Learning
[17], [18]	Patch	Kernelized Correlation filter, Dual correlation filte
[44]	Patch	Geometric, Deformable graph matching
Ours	**Keypoint**	**Classification-based SSVM+Weighted PROSAC**

by adopting a multi-task structured keypoint-model learning over several adjacent frames. Recently, [44] proposed Gracker, which is a graph-based tracker that can explore the object's structural information, improving state-of-the-art performance on many available datasets except in unconstrained environments. To this date, only a few studies have investigated on improving keypoint extraction and final pose refinement but even these focus on minimizing one particular type of errors.

Several work have adopted deep learning to estimating poses of planar objects [10], [46]. Most of these studies are performed by detecting object in each frame, requiring a very large number of images for training. For training, [35] used 600 sheets per object, and [46] used 1,000 sheets. [47] proposed a method of recognizing pose by recognizing a planar object from another viewpoint in real-time using a plurality of cameras. Nevertheless, besides the large amount of training time and data required, these trackers have only been tested on a small number of non-public custom-generated datasets, making it difficult to compare against other pipelines.

B. REMARKS ON OTHER GENERAL OBJECT TRACKERS

We briefly review some well-known tracking methods for general objects as some of them are used for comparisons in Sec. IV. [20]–[22] proposed the Tracking-Learning-Detection (TLD) structure, which builds the foundation of modern trackers for general objects. In this framework, the object model is continuously updated through growing events and is refined through pruning events which generate positive samples and negative samples respectively to update the model. Derived from this, [48] attempted to track objects by constructing multi-scale image feature space. [6], [7] proposed the kernelized correlation filter (KCF), which applies superpixels to efficiently track using color information. Henriques *et al.* proposed the dynamic graph tracker (DGT), a fast multi-channel extension of linear correlation

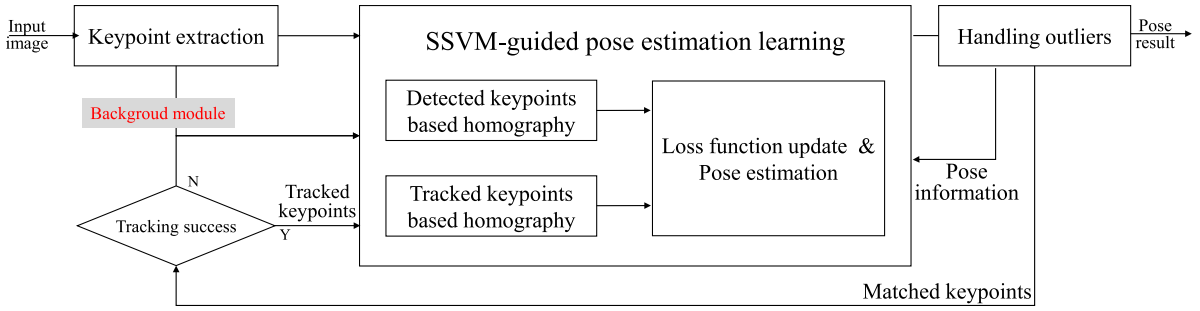


FIGURE 3. A schematic diagram of our planar object pose-estimation (POP) framework. The basis structure is similar to that of a traditional modular pipeline for planar object tracking. As mentioned in Fig. 2, we improve the keypoint extraction module by adopting grid-level dynamic thresholding (see Sec. III-A). We activate a background feature detection module for minimizing the impact of accuracy when keypoint tracking (using optical flow) fails. An SSVM-based online learning module is implemented to improve robustness against feature mismatches by learning weights for correspondences. Outliers are handled and discarded after which homography is refined.

filters using a linear kernel. [14] proposed Struck, which learns and updates the classifier using a structured support vector machine (SSVM) and estimates the object pose respectively.

III. KEY COMPONENTS IN THE POP FRAMEWORK

In this section, we illustrate our planar object pose-estimation (POP) framework in detail (see Fig. 3 for an overview). The foundation structure of POP builds upon the modular pipeline described in Sec. II. Most importantly, it comprises several additional modules and steps to take into account of 3 sources of errors described in Fig. 1.

A. KEYPOINT REFINEMENT VIA DYNAMIC THRESHOLDING

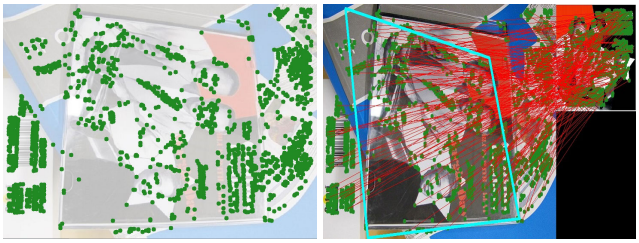
A majority of previous work [14], [15], [20], [26], [28], [49] weigh all detected keypoints equally during feature matching and homography computation. Consequently, this can result in extracting adjacent keypoints containing similar information, potentially increasing bias to an image region with high concentration of keypoints as well as raising computational burden. To resolve this, we consider the spatial distribution and quality of each keypoint, dividing image into grids and finding as-evenly-distributed-as-possible keypoints, whose qualities are higher than those of the suppressed keypoints in their respective local grids.

In each grid, we set the maximum number of keypoints (M) to a constant. If it is found that the number of keypoints (N_i) in a grid i exceeds this hard threshold, we filter and extract only the maximum number of keypoints in the grid based on the intensity corner value. If $N_i \geq M$, we proceed to the descriptor-generation stage for that grid. If $N_i < M$, then we enter the re-extraction mode for additional keypoint detection.

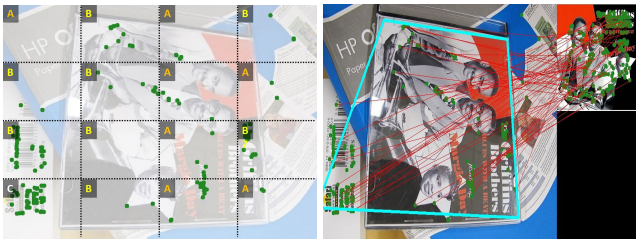
We compute M as follows: first, we set a hard threshold on the total number of keypoints allowed in an image, defining it as Q for convenience here. Then, we divide Q by the total number of grids. In terms of equations,

$$N_i \leq M = \frac{Q}{W \times H} \quad \forall \quad i \in \Omega, \tag{1}$$

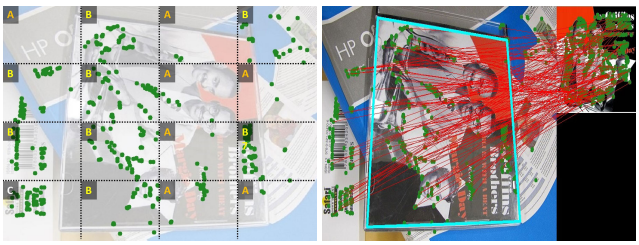
where N_i is the number of keypoints in grid i, W is the number of grids along the horizontal axis and H is the number of grids along the vertical axis (we assume rectangular grids). As we used $4 \times 4 = 16$ grids, we set the re-extraction threshold (M) as $700/16 = 43.75$. i.e. if a grid contains more the 43.75 keypoints, then we assume the grid has enough keypoints. (In such case, we store up to $1000/16 = 62.5$ keypoints.) (see Fig. 5a)



(a) Not considered

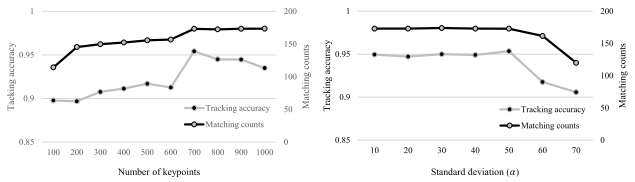


(b) Spatial distribution



(c) Proposed method

FIGURE 4. Visualization of our keypoint refinement process. (a) is a typical feature extraction result. (b) shows keypoints extracted after dividing image into grids and setting a threshold on the maximum number of keypoints in each grid. (c) shows more evenly-distributed keypoints after going through the re-extraction stage. In (b) and (c), A grids have too little pixel variations (i.e. not enough useful information to perform keypoint re-extraction), B grids go through re-extraction an C grids already have maximum number of keypoints.



(a) Accuracy vs number of keypoints (b) Accuracy vs SD threshold

FIGURE 5. Tracking accuracy as functions of different thresholds.

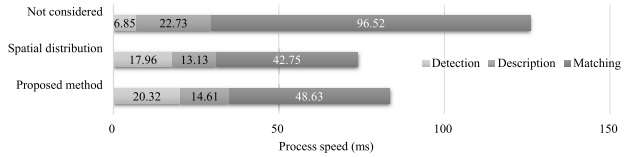


FIGURE 6. Average per-frame runtime for detection, description and matching.

During the re-extraction stage, we set the attribute of each grid by considering the standard deviation of the constituent image pixel values. If the standard deviation of pixels (σ_i) is greater than a constant threshold α, we determine the grid contains useful information and decide to extract more keypoints by lowering the utilized detector's threshold value (β). In this article, we call this process *dynamic thresholding*. On the other hand, if $\sigma_i < \alpha$, we assume the grid contains little information and quit the re-extraction process. Note that each grid is regarded as an independent attribute, meaning this stage can be processed in parallel and select keypoints relatively evenly across the image. we set α to 50 after empirically observing its slightly superior tracking accuracy over other tested pixel standard deviation (α) values (see Fig. 5b). Also, We initially set of the AGAST detector to 40. This value is lowered to 10 during every re-extraction stage.

Fig. 4 shows the advantage of our refinement module. First, Fig. 4a shows keypoints locally concentrated in specific regions of the image, increasing the bias of extracted features. Second, Fig. 4b shows that performing grid-level keypoint extraction adjusts this bias to a certain extent by compensating a large number of keypoints. Last, Fig. 4c demonstrates that re-extracting keypoints via dynamic thresholding helps to spatially even out keypoints. The addition of keypoint refinement does not necessarily lead to much increased runtime. This is partially because the module controls the number of keypoints to some limit, requiring less time during feature matching. The vanilla tracking framework (Fig. 4a) requires 126.09 ms per frame, whereas considering only spatial distribution (Fig. 4b) requires 73.84 ms and utilizing our keypoint refinement (Fig. 4c) requires 83.56 ms. Hence, the refinement module can be implemented without compensating and actually improving the real-time performance.

After going through the above-mentioned re-extraction stage, the set of extracted keypoints (K) are used for matching and pose estimation.

B. IMPROVING ROBUSTNESS TO FEATURE MISMATCHES

Matched keypoints typically contain a non-negligible portion of outlier correspondences, which need to be discarded for reliable tracking and pose estimation. We employ two modules to enhance robustness to these mismatches.

First, we activate a keypoint detection module in the background (i.e. simultaneous detection and tracking) to consistently detect for image features irrespective of optical flow-based tracking outcomes. This allows the tracker to utilize keypoints extracted from the background module when tracking fails without having to move onto the next frame.

Second, we apply a combination of the learning-based correspondence weighting along with RANSAC to improve outlier detection. Previous work have either implemented a RANSAC-variant algorithm (e.g. simple tracker) or correspondence weight learning (EOS) for outlier detection. In this work, we apply a combination of both to yield an optimal result. More specifically, for each frame, keypoint correspondences between the DB image and the query image are weighted using a structured support vector machine (SSVM), after which are used for robust homography estimation through a 4-point PROSAC [9] algorithm. The estimated candidate homographies are then fed in as training data for learning correspondence weights online. By using the updated model, we reiterate over the keypoint correspondences to recompute homography based on new weights.

1) ROBUST HOMOGRAPHY ESTIMATION

Homographies are calculated using the modified PROSAC algorithm and the matched keypoints between the query (input) image and the database (DB) image. For elaboration, let $\{K_d : \mathbf{k}_d^0, \mathbf{k}_d^1, \ldots, \mathbf{k}_d^m\}$ be a set of keypoints from the DB image and $\{K_q : \mathbf{k}_q^0, \mathbf{k}_q^1, \ldots, \mathbf{k}_q^n\}$ define a set of keypoints from the query image. We additionally define Ω as a set of matched keypoints such that $(i, j) \in \Omega$ means keypoint i from the db image match with keypoint j from the query image. Keypoint correspondences are randomly sampled multiple times from Ω, with 4 matches at a time to yield a set of candidate homographies between the DB image and the query image (we set the number of iteration 1000). The solution maximizing the number of inlier correspondences each weighted by its matching score s_{ij} is chosen as the homography matrix $\mathtt{H}$, i.e.

$$\begin{aligned}\mathtt{H} &:= \arg\max_{\mathtt{H}} f_w(\mathtt{H}) \\ &:= \arg\max_{\mathtt{H}} \sum_{(i,j)\in\Omega} s_{ij}\, \rho\left(\left\|\begin{bmatrix}\mathbf{k}_q^i\\1\end{bmatrix} - \pi\left(\mathtt{H}^{-1}\begin{bmatrix}\mathbf{k}_d^j\\1\end{bmatrix}\right)\right\|_2^2\right), \quad (2)\end{aligned}$$

$$\text{where} \quad \rho(x) = \begin{cases}1 & \text{if} \quad 0 < x < 4 \text{ pixel}^2\\ 0 & \text{otherwise}\end{cases} \quad (3)$$

is a standard RANSAC robust kernel. $\pi([x, y, z]^\mathsf{T}) := \left[\frac{x}{z}, \frac{y}{z}\right]^\mathsf{T}$ is a 3D to 2D projection function. We have designed the matching score s_{ij} to incorporate a combination of the scores outputted by the detector (e.g. FREAK) and the SSVM by defining it as

$$s_{ij} := c_{ij} \left\langle \mathbf{w}_{ij}, \mathbf{d}_j \right\rangle, \quad (4)$$

where c_{ij} is the match score from the detector, $\mathbf{d}_j$ is a *normalized* descriptor vector of keypoint i in the input image

FIGURE 7. Result of distance threshold value (a) and visualization of outlier rejection: (b) PROSAC, (c) PROSAC + inverse homography.

and $\mathbf{w}_{ij}$ is the predicted weight vector from the SSVM given a correspondence (i, j). For the first frame, $\mathbf{w}_{ij}$ is set to all-1 vector across all correspondences.

2) ONLINE LEARNING OF A STRUCTURED SUPPORT VECTOR MACHINE (SSVM) FOR LEARNING DESCRIPTOR WEIGHTS

We train a SSVM that learns to weigh keypoint descriptors given a pair of keypoint correspondences as input. For this purpose, we utilize the candidate homography matrices estimated from Sec. III-B.1. Learning is carried out using on the structured output maximum margin framework [42]. Given a pair of keypoint sets $\{K_d, K_q\}$ and a set of candidate homographies $\{\mathtt{H}\}$, the weights $\{\mathbf{w}_{ij}\}$ from Eq.(4) can be learned via a large-margin framework by solving

$$\begin{aligned}
&\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_2^2 + c\sum_{g=1}^{N}\xi_g, + \tau \sum_{(i,j)\in\Omega}\gamma_{ij},\\
&s.t.\ \xi_g \geq 0\\
&\quad \forall\, g,\ \forall\, \mathtt{H} \neq \mathtt{H}' : \delta\mathtt{H} \geq \Delta(\mathtt{H_g}, \mathtt{H})) - \xi_g\\
&\quad \forall\, i, \forall\, j\ \gamma_{ij} \geq 0\\
&\quad \forall(\mathbf{k}_d^i, \mathbf{k}_q^j), \forall j' \neq j : \left\langle \mathbf{w}_{ij}, \mathbf{d}_j - \mathbf{d}_{j'} \right\rangle \geq 1 - \gamma_{ij},
\end{aligned} \tag{5}$$

where $\mathbf{w}$ is a stacked vector of all weights $\mathbf{w}_{ij}$. $\delta\mathtt{H}$ denotes the task-dependent structured error of predicted output $\mathtt{H}$ instead of the observed output $\mathtt{H_g}$. The slack variable ξ measures the surrogate loss for the keypoints and c is the regularization parameter. The SSVM constraints force $\delta\mathtt{H}$ at $\mathtt{H} \neq \mathtt{H_g}$ to be always greater than $\delta\mathtt{H}_g$ through the slack variables $\{\xi_g\}$. In other words, $\delta\mathtt{H}$ can be interpreted as a guarantee of improved performance. As shown in Eq.(5), γ_{ij} encourages higher weights for inlier keypoint correspondences and lower weights for outliers. The loss function Δ expresses a finer distinction between $\mathtt{H_g}$ and $\mathtt{H}$, which plays an important role in the SSVM.

$$\begin{aligned}
&\Delta(\mathtt{H_g}, \mathtt{H}) = \sum_{(i,j)\in\Omega} \Psi\left(\Phi_{\mathtt{H}}, \Phi_{\mathtt{H_g}}\right)\\
&\text{where } \Phi_{\mathtt{H}} = \rho\left(\left\|\begin{bmatrix}\mathbf{k}_q^i\\1\end{bmatrix} - \pi\left(\mathtt{H}^{-1}\begin{bmatrix}\mathbf{k}_d^j\\1\end{bmatrix}\right)\right\|_2^2\right),\\
&\Psi\left(\Phi_{\mathtt{H}}, \Phi_{\mathtt{H_g}}\right) = \begin{cases}1 & \text{if } \Phi_{\mathtt{H}} \neq \Phi_{\mathtt{H_g}}\\ 0 & \text{otherwise.}\end{cases}
\end{aligned} \tag{6}$$

Eq.(5) can be solved offline with batch problem, but for real-time applications, w_{ij} should be updated online to adapt the model to a given environment. As a result, we redefine Eq.(7) to ensure real-time operation, and the learning-based optimization can be formulated as

$$\begin{aligned}
&\min_{\mathbf{w},} \frac{1}{2}\|\mathbf{w}\|_2^2 + c\sum_{g=1}^{N} err_{\mathtt{h}}, + \tau \sum_{(i,j)\in\Omega} err_d,\\
&\text{s.t. } err_{\mathtt{h}} = \max\left\{(0, \Delta(\mathtt{H_g}, \mathtt{H}) - \delta\mathtt{H})\right\}\\
&\qquad err_d = \max\left\{0, (1 - \left\langle \mathbf{w}_{ij}, \mathbf{d}_j - \mathbf{d}_{j'}\right\rangle)\right\}.
\end{aligned} \tag{7}$$

where τ is a leveraging parameter. For speed improvement, we have employed a binary descriptor for $\mathbf{d}_j$. Since SVM requires a real vector as input, we divide the descriptor into 8-bit binary numbers and convert each to a real number.

C. OUTLIER REMOVAL

Once the homography is estimated, outlier keypoints are discarded and no longer tracked. To achieve this, we check three criteria for each keypoint match, (a) keypoint error distance, (b) stability of the inlier/outlier mask and (c) stability of the computed homography. For the first criterion, we employ an inverse-homography ($\mathtt{H}^{-1}$) computation and use each matched keypoint (i) from the query (input) image to estimate the location of the corresponding keypoint (j) in the DB image (see Eq.(8),(9)). For a pair of matches (i, j), the estimated keypoint j in the DB image ($\mathbf{k}_d'^j$) from a keypoint i in the query image ($\mathbf{k}_q^i$) is obtained through the equation

$$\begin{aligned}
&\begin{bmatrix}\mathbf{k}_d'^j\\1\end{bmatrix} \simeq \mathtt{H}^{-1}\begin{bmatrix}\mathbf{k}_q^i\\1\end{bmatrix}\\
&\Rightarrow \begin{bmatrix}k_{d,x}'^j\\k_{d,y}'^j\\1\end{bmatrix} \simeq \begin{bmatrix}h_{11} & h_{12} & h_{13}\\h_{21} & h_{22} & h_{23}\\h_{31} & h_{32} & 1\end{bmatrix}^{-1}\begin{bmatrix}k_{q,x}^i\\k_{q,y}^i\\1\end{bmatrix},
\end{aligned} \tag{8}$$

$$\Rightarrow \begin{bmatrix}k_{d,x}'^j\\k_{d,y}'^j\end{bmatrix} = \begin{bmatrix}\frac{h_{11}k_{q,x}^i + h_{12}k_{q,y}^i + h_{13}}{h_{31}k_{q,x}^i + h_{32}k_{q,y}^i + 1}\\ \frac{h_{21}k_{q,x}^i + h_{22}k_{q,y}^i + h_{23}}{h_{31}k_{q,x}^i + h_{32}k_{q,y}^i + 1}\end{bmatrix} = \pi\left(\mathtt{H}^{-1}\begin{bmatrix}\mathbf{k}_q^i\\1\end{bmatrix}\right). \tag{9}$$

If the distance between the actual keypoint i and the estimated keypoint i' ($\left\|\mathbf{k}_d'^j - \mathbf{k}_d^j\right\|_2$) exceeds the threshold value of 5 pixels, the corresponding pair of matches is discarded (see Fig. 7a).

For the second criterion, we inspect the stability of the inlier/outlier mask of matched keypoints over a window of 20 frames. For each keypoint track, we record the number of times the keypoint changes from being an inlier to outlier

TABLE 2. Ablation study results of the POP framework. The top row shows the baseline tracker's mean accuracy (i.e. without any of the 3 components in Section III). Remaining rows have been generated by enabling different combinations of the modules proposed in Section III), namely keypoint refinement (***KR***), learning for improving robustness to feature mismatches (***LIR***) and outlier handling for homography estimation (***OH***).

	EOS	MTSC	LinTrack	MSL	UCSB	**Avg.**
Baseline	65.9 ± 32.7	59.4 ± 22.4	47.7 ± 29.1	35.8 ± 15.8	50.5 ± 21.4	52.5 ± 26.7
KR *only*	82.7 ± 18.7	92.4 ± 10.1	85.1 ± 12.6	75.7 ± 31.6	59.8 ± 31.6	69.1 ± 20.9
LIR *only*	88.0 ± 20.6	92.5 ± 11.2	85.5 ± 14.0	79.5 ± 33.5	80.5 ± 24.4	75.2 ± 20.7
OH *only*	77.7 ± 18.8	93.4 ± 8.9	79.1 ± 17.6	78.3 ± 32.3	78.9 ± 20.1	71.5 ± 19.5
KR + ***LIR***	93.9 ± 6.7	95.6 ± 3.6	86.3 ± 18.0	85.0 ± 39.1	86.1 ± 19.7	80.0 ± 17.4
KR + ***OH***	94.6 ± 8.9	95.4 ± 6.1	89.1 ± 9.6	82.1 ± 37.2	81.0 ± 21.4	78.8 ± 16.6
LIR + ***OH***	94.7 ± 7.0	95.7 ± 4.4	88.6 ± 10.7	85.0 ± 38.3	81.2 ± 22.2	79.4 ± 16.5
POP	**95.8 ± 1.7**	**95.8 ± 1.4**	**94.8 ± 1.2**	**93.2 ± 3.3**	**89.1 ± 4.9**	**93.1 ± 4.2**

TABLE 3. Average accuracies (%) and per-frame runtimes (s) of planar object trackers on various benchmarks. Our AGAST-FREAK-implemented POP algorithm shows the highest accuracy with smallest standard deviation across all tested datasets.

	EOS	MTSC	LinTrack	MSL	UCSB	**Avg.**	**Time (s)**
Baseline	65.9 ±32.7	59.4 ±22.4	47.7 ±29.1	35.8 ±15.8	50.5 ±21.4	52.5 ±26.7	0.023
Struck [14]	12.8 ±7.2	31.2 ±17.9	27.5 ±20.3	28.5 ±8.4	47.8 ±25.3	31.3 ±22.1	0.098
EOS [15]	91.9 ±6.5	61.3 ±14.4	82.3 ±11.2	87.7 ±16.1	62.2 ±17.7	75.6 ±19.5	0.031
GPF [25]	70.9 ±37.3	**98.1 ±0.5**	15.2 ±12.0	84.4 ±21.2	44.2 ±23.8	62.6 ±35.8	0.048
DGT [6], [7]	18.5 ±10.4	25.4 ±8.6	46.2 ±29.9	20.6 ±5.7	42.2 ±8.4	30.9 ±17.5	0.320
KCF [17], [18]	67.0 ±19.6	51.0 ±27.3	87.8 ±11.7	84.1 ±4.7	77.8 ±21.5	73.2 ±22.9	0.009
Gracker [44]	94.8 ±2.9	95.8 ±2.3	93.6 ±3.9	**92.6 ±2.5**	84.9 ±12.4	91.4 ±8.5	0.144
Ours	**95.8 ±1.7**	95.8 ±1.4	**94.8 ±1.2**	**93.2 ±3.3**	**89.1 ±4.9**	**93.1 ±4.2**	0.114

TABLE 4. Tracking success rates for different combinations of feature detector and descriptor applied to our POP framework. AGAST+FREAK shows the best accuracy and are used for comparing against other planar object tracking algorithms in Sec. IV.

Method		Accuracy (%)
Detector	Descriptor	
FAST	BRIEF	95.37 ±13.89
	BRISK	95.37 ±13.89
	SURF	95.37 ±13.89
	FREAK	95.34 ±13.87
AGAST	BRIEF	95.39 ±13.82
	BRISK	84.33 ±34.34
	SURF	62.03 ±48.29
	FREAK	**99.92 ± 0.21**
Hessian	BRIEF	95.39 ±13.83
	BRISK	95.39 ±13.83
	SURF	95.39 ±13.83
	FREAK	95.37 ±13.89
CenSure	BRIEF	95.41 ±13.78
	BRISK	91.19 ±17.51
	SURF	95.41 ±13.78
	FREAK	84.28 ±34.36
Mean tracking accuracy		91.93 ± 17.94

TABLE 5. Tracking success rates for each detector and descriptor applied to our POP framework.

Detector	Accuracy (%)	Descriptor	Accuracy (%)
FAST	95.36 ±13.89	BRIEF	95.39 ±13.83
AGAST	85.42 ±24.17	BRISK	91.57 ±19.89
Hessian	95.39 ±13.85	SURF	87.05 ±22.45
Censure	91.57 ±19.86	FREAK	93.73 ±15.58

TABLE 6. Average tracking accuracies (%) on the TMT dataset. Our POP algorithm shows the better accuracy than Gracker in TMT dataset.

object	variation	Gracker	**Ours**
book I	tilting	**96.2 ±2.4**	95.6 ±2.3
book II	zoom	**98.2 ±0.5**	95.7 ±1.9
book III	occlusion	93.2 ±2.8	**95.8 ±2.3**
cereal	rotation	91.2 ±10.1	**92.7 ±3.2**
juice	rotation	90.3 ±11.6	**94.1 ±2.1**
mug I	translation	93.1 ±1.5	**96.3 ±2.2**
mug II	tilting	78.9 ±6.4	**92.2 ±2.0**
mug III	rotation	82.9 ±5.2	**94.4 ±2.6**
composite	unconstrained	87.7 ±13.2	**93.8 ±3.1**
Mean tracking accuracy		90.4 ±9.4	**94.5 ± 2.8**

TABLE 7. Average tracking accuracies (%) on the POT210 dataset. Our POP algorithm shows the better accuracy than Gracker in POT210 dataset.

	Gracker	**Ours**
Scale change	74.9 ±24.8	**84.7 ±14.6**
Rotation	82.1 ±14.2	**83.7 ±10.1**
Perspective distortion	**87.5 ±12.6**	79.4 ±14.9
Motion blur	24.0 ±10.0	**53.5 ±21.4**
Occlusion	**90.9 ±7.5**	83.6 ±14.2
Out of view	**92.9 ±3.5**	88.4 ±9.7
Unconstrained	64.6 ±13.7	**83.4 ±13.7**
Mean tracking accuracy	73.8 ±26.2	**79.6 ±18.2**

or vice versa. Any keypoint track switching more than 5 times is regarded unstable and removed. If this results in fewer than 4 inlier point tracks (i.e. minimum number of points required to compute a homography), we utilize the background detection module's matched keypoints to recompute homography and discard outlier keypoints based on the first criterion only. This has an effect of filling up a new pool of keypoint tracks.

For the last criterion, we check the smoothness of the tracked object's pose by observing the magnitude of drift of the object vertices. The 4 vertices of the planar object are retrieved from the DB image and then projected to the query image. Assuming similar projections of the vertices have been made for the previous frame $q-1$, we compute the sum of vertex movements as follows:

$$\sum_{i=1}^{4} \left\| \pi\left(\mathtt{H}_q \begin{bmatrix} \mathbf{k}_d^j \\ 1 \end{bmatrix} \right) - \pi\left(\mathtt{H}_{q-1} \begin{bmatrix} \mathbf{k}_d^j \\ 1 \end{bmatrix} \right) \right\|_2^2, \tag{10}$$

where j denotes a vertex keypoint, and $\mathtt{H}_q$ and $\mathtt{H}_{q-1}$ are the homography matrices for the current query image and the previous query image respectively. If the square root of Eq.(10) is above 30 pixels, we determine the homography is unstable. Additionally, if the projected vertices (in the query image) are not in the correct order (due to a twisted

TABLE 8. Average tracking accuracies. Each row represents a dataset and each column represents an algorithm.

	Baseline	Struck	EOS	GPF	DGT	KCF	Gracker	**POP**
EOS								
Barbapapa	92.4 ± 23.8	5.3 ± 21.7	93.8 ± 21.1	52.4 ± 47.1	30.0 ± 29.4	68.1 ± 42.9	89.3 ± 24.3	93.5 ± 11.4
Comic	64.1 ± 46.9	9.8 ± 24.1	94.7 ± 15.6	97.9 ± 1.2	5.5 ± 12.7	82.6 ± 8.2	95.7 ± 1.5	95.3 ± 1.6
Map	82.9 ± 36.7	8.3 ± 20.7	97.7 ± 11.0	99.6 ± 0.4	9.6 ± 16.7	72.2 ± 14.0	97.0 ± 1.3	98.4 ± 1.4
Paper	3.3 ± 17.8	14.9 ± 36.3	79.1 ± 38.2	5.6 ± 20.7	30.8 ± 25.3	29.6 ± 45.2	94.4 ± 3.1	94.3 ± 2.2
Phone	87.0 ± 30.3	25.7 ± 28.9	94.1 ± 18.6	98.9 ± 6.5	16.4 ± 17.1	82.5 ± 14.9	97.6 ± 1.0	94.6 ± 4.7
MTSC								
Book	70.0 ± 42.2	10.1 ± 25.4	51.7 ± 47.3	98.1 ± 1.4	25.0 ± 27.1	8.2 ± 20.3	91.8 ± 2.9	94.9 ± 3.0
Chart	58.1 ± 47.5	23.4 ± 32.7	52.8 ± 48.7	97.3 ± 1.6	11.5 ± 16.2	83.8 ± 20.2	96.5 ± 3.9	95.2 ± 3.0
Food	85.1 ± 34.9	59.1 ± 23.3	86.1 ± 33.3	98.8 ± 1.0	33.0 ± 11.1	52.6 ± 18.9	97.9 ± 0.8	98.2 ± 1.6
Keyboard	24.3 ± 42.6	32.0 ± 33.4	54.6 ± 48.7	98.3 ± 1.5	32.1 ± 20.5	59.3 ± 22.4	96.9 ± 1.6	95.0 ± 2.4
LinTrack								
Mousepad	83.8 ± 34.5	16.1 ± 28.5	97.8 ± 0.9	31.8 ± 15.0	24.0 ± 40.1	98.5 ± 3.1	96.5 ± 4.7	96.4 ± 4.0
Phone	12.4 ± 32.5	10.4 ± 29.5	77.4 ± 38.1	10.1 ± 29.1	26.1 ± 37.0	93.4 ± 6.9	88.1 ± 20.6	93.4 ± 3.6
Towel	47.0 ± 45.1	56.0 ± 18.9	71.6 ± 36.2	3.8 ± 16.1	88.5 ± 21.7	71.5 ± 17.6	96.2 ± 1.6	94.7 ± 8.7
MSL								
Painting	59.7 ± 48.3	28.7 ± 29.9	59.9 ± 48.0	97.7 ± 2.0	26.0 ± 23.1	87.4 ± 7.1	92.4 ± 3.3	90.4 ± 6.9
Graph	35.1 ± 47.4	42.3 ± 28.6	99.3 ± 0.8	47.7 ± 38.5	22.0 ± 16.3	88.7 ± 8.2	94.5 ± 2.1	98.9 ± 1.3
Book title	15.1 ± 34.3	21.8 ± 26.6	95.6 ± 3.7	95.9 ± 2.8	23.2 ± 16.9	76.7 ± 34.5	88.5 ± 4.1	91.4 ± 4.0
Logo	33.5 ± 46.0	21.4 ± 25.0	96.2 ± 12.7	96.2 ± 12.6	10.9 ± 15.1	83.7 ± 18.2	94.9 ± 1.7	91.9 ± 4.1
UCSB								
Lighting	81.4 ± 18.6	93.4 ± 1.3	77.4 ± 18.1	91.0 ± 1.8	43.4 ± 17.2	92.9 ± 3.6	99.3 ± 0.3	88.9 ± 12.0
Motion blur	34.4 ± 31.1	24.7 ± 36.0	32.1 ± 30.6	31.7 ± 37.9	30.8 ± 29.1	88.8 ± 7.0	65.9 ± 13.0	80.1 ± 16.4
Panning	75.9 ± 28.9	25.6 ± 37.5	68.9 ± 17.2	19.8 ± 25.9	45.3 ± 21.9	97.0 ± 2.6	88.9 ± 4.9	94.6 ± 4.4
Perspective	62.6 ± 46.2	58.9 ± 46.5	76.7 ± 38.2	57.2 ± 46.6	45.6 ± 33.6	61.4 ± 47.1	85.0 ± 25.0	90.7 ± 9.5
Rotation	31.1 ± 44.5	71.2 ± 21.1	68.0 ± 23.3	55.4 ± 19.5	46.5 ± 22.9	70.7 ± 20.1	95.6 ± 2.4	91.1 ± 8.8
Unconstrained	22.2 ± 39.3	23.5 ± 39.2	37.7 ± 42.0	18.4 ± 37.1	29.3 ± 44.3	35.6 ± 44.3	67.0 ± 29.8	84.2 ± 8.9
Zoom	46.1 ± 45.6	37.6 ± 43.8	74.9 ± 15.7	35.6 ± 46.3	54.6 ± 35.9	98.4 ± 2.2	92.8 ± 3.5	94.2 ± 3.6

or concave motion) or concentrated in a very small region, the corresponding homography is also discarded. In these unstable-determined cases, we fall back to the homography estimated from the previous frame.

IV. EXPERIMENTAL RESULTS

We conducted 4 experiments to analyze the performance of our POP framework in detail. First, we conducted an ablation study to empirically show the performance gain brought by each stage of the framework and indicate the need for all the stages introduced to achieve top performance. Second, we compared the tracking accuracies achieved by different combinations of feature detectors and descriptors when using POP to demonstrate the generic nature of the framework. Third, we compared our winning POP combination from the second experiment against other baseline and state-of-the-art planar tracking algorithms on various datasets, empirically demonstrating the scheme's robustness to unconstrained tracking environments. Last, we selected the two best performing algorithms from the 3rd experiment and compared their accuracies on more challenging datasets (TMT and POT210) comprising various unconstrained tracking environments. All the experiments were conducted on a PC with an Intel Core i7-4790 (3.6 GHz) CPU and 16 GB RAM.

Datasets We tested our method on 6 public datasets and 1 custom dataset, namely UCSB (96 sequences, 6,889 frames) by [13], LinTrack (3 sequences, 12,477 frames) by [50], EOS (5 sequences, 4,001 frames) by [15], MTSC (4 sequences, 2,731 frames) by [49], TMT (109 sequences, 60,254 frames) by [41] and POT210 (210 sequences, 105,000 frames) by [30] and MSL (our own, 4 sequences, 4,175 frames). We did not use VOT [24] and OTB [45] benchmarks as these are for tracking common objects including non-planar and non-rigid bodies. Moreover, annotations in these benchmarks are given as axis-aligned bounding boxes rather than as exact regions, making them inappropriate for evaluating planar object trackers [44].

A. ABLATION STUDY

To empirically observe how each component in Section III affects the tracking performance, we tested all $2^3 = 8$ combinations, which were created by disabling different sets of the 3 proposed modules. Corresponding results are illustrated in Table 2. From the Table 2, we verify adding each component to the baseline planar object tracker improves mean tracking accuracy, justifying the need for each stage in POP.

B. COMPARING DIFFERENT COMBINATIONS OF FEATURE DETECTORS AND DESCRIPTIONS

In the second experiment, we compared different combinations of feature detectors and descriptors applied to our POP scheme on EOS and MTSC benchmark datasets. For feature detectors, we applied accelerated segment test (FAST) [40], AGAST [33], Hessian [34], and center surround extrema (CenSure) [1]. For feature descriptors, we included

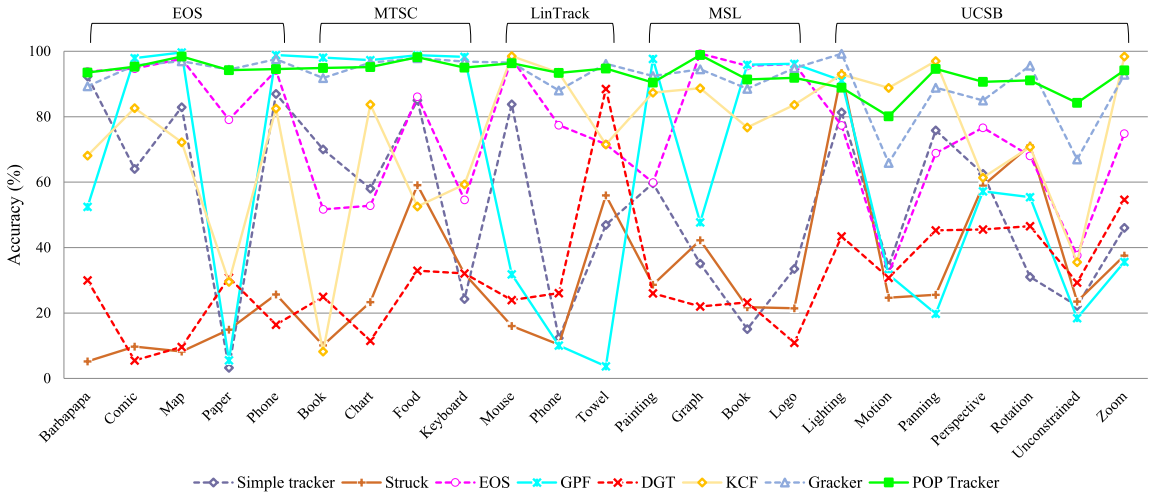


FIGURE 8. Comparison of tracking accuracies achieved by various planar object tracking algorithms. Datasets in the horizontal axis are aligned in the ascending order of difficulty. Our POP tracker (AGAST-FREAK) in green achieves similar performance to state-of-the-art trackers on easier baseline datasets, but it consistently achieves relatively high accuracy on more difficult ones. Note POP is the only algorithm achieving over 80% across all tested datasets.

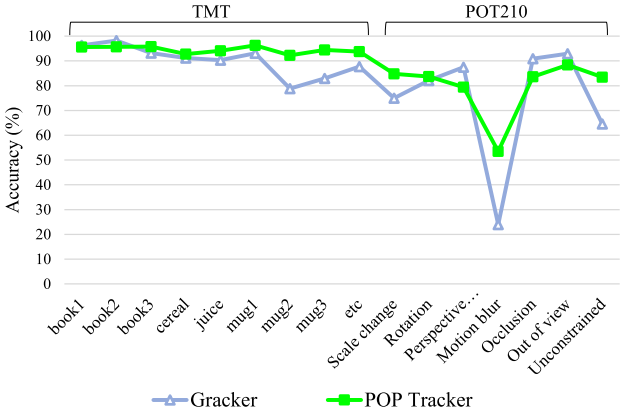


FIGURE 9. Comparison of tracking accuracies achieved by Gracker and our POP algorithms.

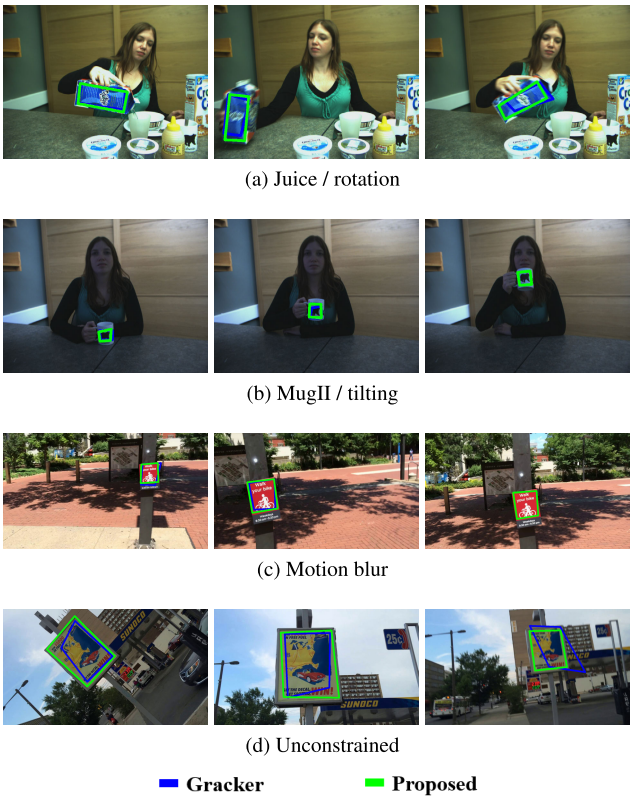


FIGURE 10. Sample results comparing our POP (AGAST-FREAK) algorithm against Grackers. POP recovers comparatively accurate regions of interest (ROIs) in challenging environments.

binary robust independent elementary features (BRIEF) [8], binary robust invariant scalable keypoints (BRISK) [29], speeded-up robust features (SURF) [3] and fast retina keypoint (FREAK) [43].

For measuring tracking success rate, we reported the proportion of frames satisfying two criteria. First, the fraction of matched keypoints over total number of keypoints must exceed η, which we set to 20%. Second, the value of the scoring function $S(\hat{\mathrm{H}}, \mathrm{H})$ has to be smaller than the threshold value T_d set to 10. We used [15]'s scoring function defined as

$$S(\hat{\mathrm{H}}, \mathrm{H}) := \sum_{k=1}^{N} \left\| \mathrm{H}\tilde{\mathbf{c}}_k - \hat{\mathrm{H}}\tilde{\mathbf{c}}_k \right\|_2^2, \tag{11}$$

where $\hat{\mathrm{H}} \in \mathbb{R}^{3\times3}$ is the ground truth homograph matrix, $\mathrm{H} \in \mathbb{R}^{3\times3}$ is the predicted homography matrix [15] and $\{\tilde{\mathbf{c}}_k\} \subset \mathbb{R}^3$ are the homogeneous vertices of a square of length 2 centered at the origin (for the purpose of normalization). We reported the number of frames with $S(\hat{\mathrm{H}}, \mathrm{H}) < T_d$.

Table 4 summarizes the tracking accuracies achieved by each combination. AGAST-FREAK achieved the best tracking success rate by reaching 99.92% while AGAST-SURF pair showed the worst result of 62.03%. Looking at the average accuracy of each detector and descriptor, AGAST performs the best across all detectors and BRIEF achieves top performance amongst tested descriptors. (see Table. 5). The average tracking accuracy achieved across all combinations is 91.93%, indicating our POP framework is generic and not

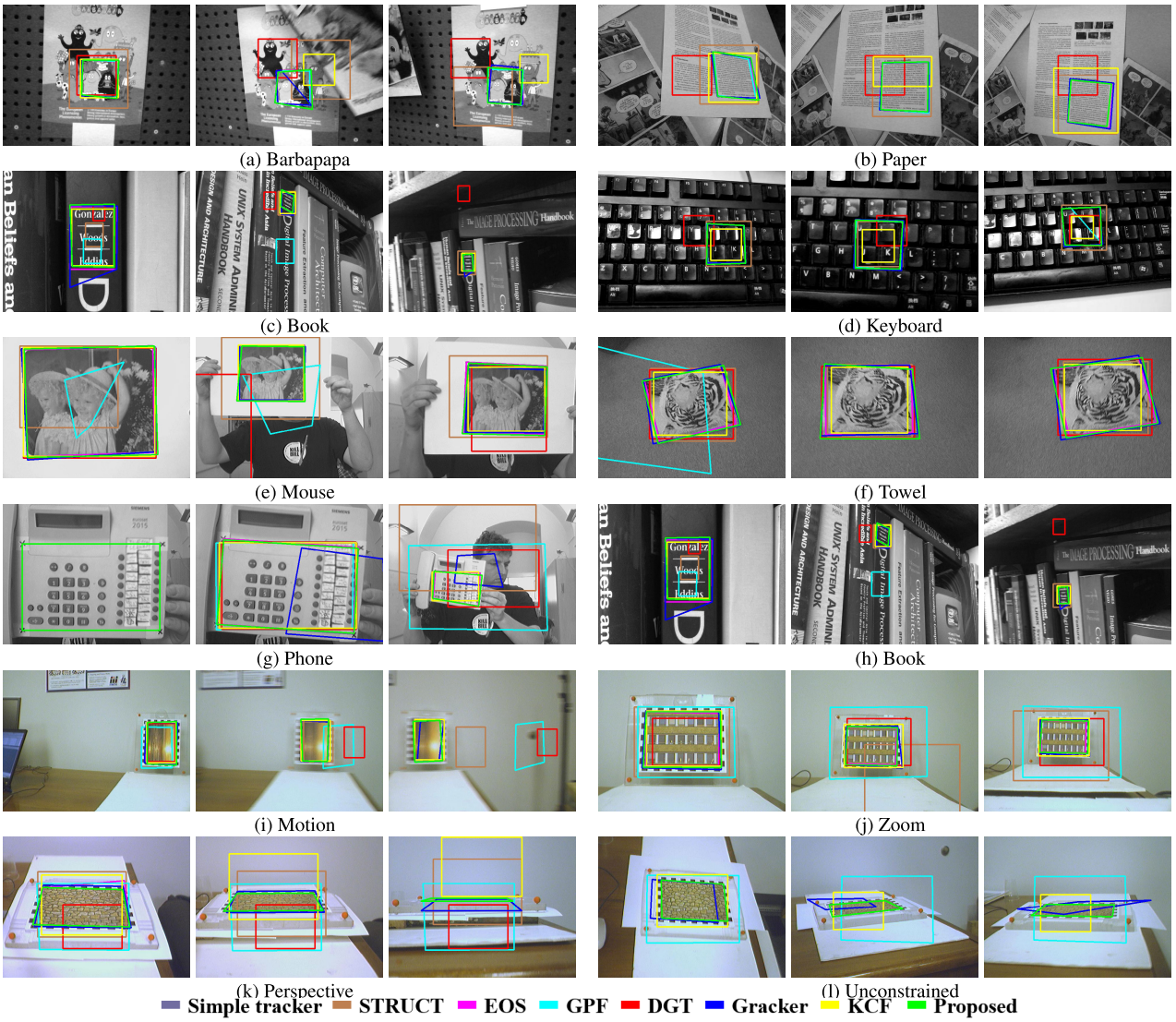


FIGURE 11. Sample results comparing our POP (AGAST-FREAK) algorithm against other baseline and state-of-the-art plane object trackers. POP recovers comparatively accurate regions of interest (ROIs) in versatile environments. For *perspective* and *unconstrained*, our POP is the only one finding the correct ROIs. In some datasets, some algorithms fail to grasp the ROIs.

necessarily dependent on a particular selection of detector and descriptor.

C. TRACKING ACCURACY AND RUNTIME

In the third experiment, we compared our winning AGAST-FREAK-equipped POP algorithm against other baseline and state-of-the-art planar tracking algorithms, namely EOS [15], GPF [25], Gracker [44], adaptive generic accelerated segment test (AGAST) [33]-based baseline, Struck [14], DGT [6], [7] and KCF [17], [18]. The first 4 are planar object-specific trackers while the last 3 are for general objects (these 3 are sometimes used for evaluation purposes in planar object tracking [44]).

For measuring the tracking accuracy, we used a region-of-interest (ROI) overlap ratio [12], which computes the proportion of area correctly detected by tracker with respect to ground truth. This is a widely used metric for comparing the pose estimation accuracy [44]. For UCSB datasets, we used the area within the black-and-white frame as ground truth. For others, we used ground truth regions provided by the dataset authors.

Table 3, Table 8 (detailed pose estimation results), Fig. 8 and Fig. 11 show that our POP tracker overall outperforms other trackers by achieving best tracking accuracy on most benchmarks when compared to other algorithms. GPF, EOS, and DGT confirmed failed more frequently during fast motion or rapid scale changes. In particular, GPF and DGT showed weaknesses for fast movement and rotation.

For most benchmarks, we found that our POP tracker and Gracker showed comparatively high accuracies. As shown in Fig. 11, most of the baseline algorithms were vulnerable to scenes with strong perceptiveness. Gracker especially showed worsened performance as the viewpoint angle increased. DGT, and GPF showed significantly lower accuracies in unconstrained versatile scene conditions. KCF showed

weaknesses for datasets with occlusions. This is most likely to have been caused by the algorithm's confusion due to similar texture around the neighborhood.

Our real-time performance comparisons can be found in Table 3 and Table 8. The fastest runtime is achieved by KCF, although its accuracy shows large variations. On the other hand, the slowest object tracker was DGT which also showed poor accuracies. Gracker showed similar but slightly slower runtime than ours. POP took the third longest time out of the tested algorithms, but its accuracy is overall consistently high and still can be run in real-time across all datasets (see Table 3 and Table 8).

D. TRACKING ACCURACIES IN TMT AND POT210

In the last experiment, we selected the two best performing algorithms from the 3rd experiment (see Section IV-C), namely Gracker and POP, and compared their performances on more challenging datasets, namely TMT and POT210. The TMT dataset consists of the sequences named *tilt*, *zoom*, *occlusion*, *rotation*, *translation* and *unconstrained*, and the POT210 dataset comprises *scale change*, *rotation*, *perspective distortion*, *motion blur*, *occlusion*, *out-of-view*, and *unconstrained*.

Table 6, Table 7, Fig. 9 and Fig. 10 show that the POP-based tracker overall outperforms Gracker. POP achieved higher accuracies in 7 out of 9 sequences in TMT. POP's mean accuracy was 4.1% higher than that of Gracker (see Table 6).

On the POT210 dataset, POP showed better accuracies in 4 out of 7 sequences, with mean accuracy 5.8 % above that of Gracker. The reason for such large difference in the mean accuracy is that POP showed relatively consistent results across the sequences while Gracker showed extremely poor performance on the *motion blur* and *unconstrained* sequences. We believe the POP's ability to recover from tracking failures through parallel detection and tracking mechanism as well as dynamic thresholding of grid-level keypoint detection helps to produce consistent results on challenging scenes (see Table 7 for details).

V. CONCLUSION

In this article, we have proposed POP, a generic planar object pose-estimation framework for mitigating several correlated sources of errors arising in unconstrained environments. More specifically, we introduced a keypoint refinement step to improve quality of keypoints, added a detection module working in background as well as an online learning framework for more accurate homography estimation to reduce feature matching errors and attached an outlier removal step to minimize errors from homography estimation. Through extensive experimental comparisons, we empirically demonstrated the effectiveness of POP in various situations including unconstrained environments. We believe this suggests a need to consider all sources of errors simultaneously to achieve stable performance in such dynamic scene.

Future work will focus on improving each added module in POP, for instance introducing a learning technique for pose estimation or transferring advantages of other tracking algorithms. We will also investigate on reducing errors arising from unreliable keypoint tracking, aiming towards reaching the full potential of our POP in versatile AR environments.

REFERENCES

[1] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2008, pp. 102–115.

[2] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 983–990.

[3] H. Bay, T. Tuytelaars, and L. Gool, "Surf: Speeded up robust features," in *Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2006, pp. 404–417.

[4] V. Beglov, *Object Information Based on Marker Recognition*. Kuopio, Finland: Univ. Eastern Finland, 2013.

[5] W. Bouachir and G.-A. Bilodeau, "Structure-aware keypoint tracking for partial occlusion handling," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2014, pp. 877–884.

[6] Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, and S. Z. Li, "Robust deformable and occluded object tracking with dynamic graph," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5497–5509, Dec. 2014.

[7] Z. Cai, L. Wen, J. Yang, Z. Lei, and S. Z. Li, "Structured visual tracking with dynamic graph," in *Proc. Asian Conf. Comput. Vis. (ACCV)*. Springer, 2012, pp. 86–97.

[8] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2010, pp. 778–792.

[9] O. Chum and J. Matas, "Matching with PROSAC—Progressive sample consensus," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 220–226.

[10] T.-T. Do, M. Cai, T. Pham, and I. Reid, "Deep-6DPose: Recovering 6D object pose from a single RGB image," 2018, *arXiv:1802.10367*. [Online]. Available: http://arxiv.org/abs/1802.10367

[11] D. Du, L. Wen, H. Qi, Q. Huang, Q. Tian, and S. Lyu, "Iterative graph seeking for object tracking," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1809–1821, Apr. 2018.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[13] S. Gauglitz, T. Höllerer, and M. Turk, "Evaluation of interest point detectors and feature descriptors for visual tracking," *Int. J. Comput. Vis.*, vol. 94, no. 3, p. 335, Sep. 2011.

[14] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.

[15] S. Hare, A. Saffari, and P. H. S. Torr, "Efficient online structured output learning for keypoint-based object tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1894–1901.

[16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[17] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2012, pp. 702–715.

[18] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[19] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D object pose estimation," 2018, *arXiv:1808.08319*. [Online]. Available: http://arxiv.org/abs/1808.08319

[20] Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops, ICCV Workshops*, Sep. 2009, pp. 1417–1424.

[21] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 49–56.

[22] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.
[23] H. Kato and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmented reality conferencing system," in *Proc. 2nd IEEE ACM Int. Workshop Augmented Reality (IWAR)*, 1999, pp. 85–94.
[24] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder, "The visual object tracking vot2015 challenge results," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Dec. 2015, pp. 1–23.
[25] J. Kwon, H. S. Lee, F. C. Park, and K. M. Lee, "A geometric particle filter for template-based visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 625–643, Apr. 2014.
[26] J. Kwon and K. M. Lee, "Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping Monte Carlo sampling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1208–1215.
[27] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1269–1276.
[28] J. Kwon and K. Mu Lee, "Tracking by sampling trackers," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1195–1202.
[29] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2548–2555.
[30] P. Liang, Y. Wu, H. Lu, L. Wang, C. Liao, and H. Ling, "Planar object tracking in the wild: A benchmark," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 651–658.
[31] J. Lim, D. A. Ross, R.-S. Lin, and M.-H. Yang, "Incremental learning for visual tracking," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 793–800.
[32] B. D. Lukas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Image Understand. Workshop*, 1981.
[33] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2010, pp. 183–196.
[34] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proc. 8th IEEE Int. Conf. Comput. Vis. ICCV*, Jul. 2001, pp. 525–531.
[35] Y. Nakajima and H. Saito, "Robust camera pose estimation by viewpoint classification using deep learning," *Comput. Vis. Media*, vol. 3, no. 2, pp. 189–198, Jun. 2017.
[36] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–4302.
[37] U. Neumann and S. You, "Natural feature tracking for augmented reality," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 53–64, Mar. 1999.
[38] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 427–436.
[39] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, May 2008.
[40] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2006, pp. 430–443.
[41] A. Roy, X. Zhang, N. Wolleb, C. P. Quintero, and M. Jagersand, "Tracking benchmark and evaluation for manipulation tasks," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 2448–2453.
[42] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *J. Mach. Learn. Res.*, vol. 6, pp. 1453–1484, Sep. 2005.
[43] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 510–517.
[44] T. Wang and H. Ling, "Gracker: A graph-based planar object tracker," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1494–1501, Jun. 2018.
[45] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.
[46] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*. [Online]. Available: http://arxiv.org/abs/1711.00199
[47] L.-W. Xue, L.-G. Chen, J.-Z. Liu, Y.-J. Wang, Q. Shen, and H.-B. Huang, "Object recognition and pose estimation base on deep learning," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2017, pp. 1288–1293.
[48] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Springer, 2012, pp. 864–877.
[49] L. Zhao, X. Li, J. Xiao, F. Wu, and Y. Zhuang, "Metric learning driven multi-task structured output optimization for robust keypoint tracking," in *Proc. AAAI*, 2015, pp. 3864–3870.
[50] K. Zimmermann, J. Matas, and T. Svoboda, "Tracking by an optimal sequence of linear predictors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 4, pp. 677–692, Apr. 2009.

SEUNGHO CHAE received the Ph.D. degree in computer science from Yonsei University, Seoul, South Korea, in 2018. He is currently a Postdoctoral Researcher with the Korea Institute of Science and Technology (KIST). His research interests include computer vision, augmented reality, and human–computer interaction. Particularly, his research focuses on object tracking and person re-identification.

JE HYEONG HONG received the M.Eng. degree in electrical and information sciences and the Ph.D. degree in engineering from the University of Cambridge, U.K., in 2011 and 2018, respectively. He is currently serving military-replacement service as a Postdoctoral Researcher with the Korea Institute of Science and Technology (KIST). His research interests include optimization problems and methods in electronics and computer vision.

HEESEUNG CHOI received the B.S., M.S., and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2004, 2006, and 2011, respectively. He had been a Research Member of the Biometrics Engineering Research Center (BERC), South Korea, and computer science and engineering from Michigan State University, USA. He is currently a Research Member with the Center for Imaging Media Research (CIMR), Korea Institute of Science and Technology (KIST). His research interests include computer vision, biometrics, image processing, forensic science, and pattern recognition.

IG-JAE KIM (Member, IEEE) received the B.S. and M.S. degrees in EE from Yonsei University, Seoul, South Korea, in 1996 and 1998, respectively, and the Ph.D. degree from Seoul National University, in 2009. He was with the Massachusetts Institute of Technology (MIT) Media Laboratory as a Postdoctoral Researcher from 2009 to 2010. He is currently the Director of the Center for Imaging Media Research, Korea Institute of Science and Technology (KIST), Seoul. He is also an Associate Professor with the KIST School, University of Science and Technology (UST). He has published more than 80 fully referred articles in international journal and conferences, including the *ACM Transactions on Graphics*, SIGGRAPH, *Pattern Recognition*, and ESWA. His current research interests include pattern recognition, computer vision, computer graphics, and computational photography.