Intelligent Automation And Soft Computing, 2019
Copyright © 2019, TSI® Press
Vol.25, no.1, 205–215

Visual Object Detection and Tracking using Analytical Learning Approach of Validity Level

Yong-Hwan Lee, Hyochang Ahn, Hyo-Beom Ahn, Sun-Young Lee

Dept. of Digital Contents, Wonkwang University, Iksan, Jeonbuk, Korea
Far East University, Eumseong, Chungbuk, Korea
College of Engineering, Kongju National University, Cheonan, Chungnam, Korea
Dept. of Information Security Engineering, Soonchunhyang University, Asan-si, Chungnam, Korea

ABSTRACT
Object tracking plays an important role in many vision applications. This paper proposes a novel and robust object detection and tracking method to localize and track a visual object in video stream. The proposed method is consisted of three modules; object detection, tracking and learning. Detection module finds and localizes all apparent objects, corrects the tracker if necessary. Tracking module follows the interest object by every frame of sequences. Learning module estimates a detecting error, and updates its value of credibility level. With a validity level where the tracking is failed on tracing the learned object, detection module finds again the desired object. The experimental results show that the proposed approach is more robust in appearance changes, viewpoint changes, and rotation of the object, compared to the traditional method. The proposed method can track the interest object accurately in various environments.

KEY WORDS: Analytical Learning Approach, Camshift, Object Detection, Object Tracking, SURF (Speeded-Up Robust Feature), Validity Level

1 INTRODUCTION

SINCE object tracking plays an important role and has become a very popular problem in the field of computer vision, many tracking systems have been proposed and developed to solve the problem both in academia and industry (Hanxuan, 2011). Currently, as the market of mobile devices are expanding with the proliferation of high-powered mobile computing environments, the availability of high quality video and the increasing need for automated video analysis make a great interest in object tracking algorithms. The object tracking has widely application in many areas, such as motion-based recognition, automated surveillance, security, video indexing, traffic monitoring, automobile navigation, and gesture recognition of human computer interaction (Suraj, 2016). Let’s consider an analysis of the video stream taken by built-in camera of mobile device depicting various objects moving into and out of the camera view. Given a bounding box defining an interest object in video frames, we need to determine a boundary of the object, and to indicate whether the object is in the frames or not. This process is done by three main steps; (1) video analysis, which detects movement of the interesting objects, (2) tracking the objects from frame to frame, and (3) analysis of the object tracking to recognize their behaviors (Alper, 2006). In an object detecting from video sequences, the object is defined as anything interesting for analysis. For example, people walking on a road, cars in the road, animals in the field, and so on. The form of appearance object is classified into three types, as the locating the position by points, expressing the area by bounding boxes, and drawing the object contours (Xi, 2013). Selecting good feature plays a critical role in the object detecting and tracking, and feature selection is closely related to the object representation (Jianbo, 1994). Many approaches for object detection use a combination of the features, such as color, shape, geometry, and so on. In an object tracking, tracking is an estimation and analysis of trajectories of the object in the frame by moving through video sequences (Zdenek, 2012). Many approaches for tracking use one of main existing algorithm, as block-matching, KLT (Kanade-Lucas-Tomasi) algorithm (Jianbo, 1994), Kalman filter (Dorin, 2003), MeanShift (Diansheng, 2009) or CamShift (Kenji, 2010).

CONTACT Sun-Young Lee ✉ sunlee@sch.ac.kr

This paper proposes an efficient real-time detection and tracking method of visual object using TLD (Tracking, Learning and Detecting) approach. The main contribution of this paper is a novel design of the TLD method that decomposes visual object tracking task into three sub-tasks, which are object detecting, tracking and learning. Each sub-task is addressed by a single module, and the modules are operated simultaneously. *Detecting* module finds and localizes all apparent objects that have been observed, and corrects the tracker if necessary. *Tracking* module follows the interest object from frame to frame. *Learning* module estimates a detecting error, and updates its value of credibility level. As under valid level which means that the tracking is failed on trace the learned object, *detecting* module finds again the desired object. Such as in the case of surveillance, moving object is usually detected and tracked in the static camera (Enrico, 2016). This means that the application has a limit of the location from the stationary camera. Since mobile devices are used in a variety of applications, this paper performs moving object tracking under moving camera. In this case, failure of object tracking may occur more frequently, because of covering the tracked object with another object and/or shaking of the moving camera. Thus, re-detecting the desired object is important when the tracked object is missing. To efficiently detect again and trace the missing object, adaptive learning process is necessary to update the tracked object model to avoid any missing track. That is why learning module plays an important role in the proposed scheme.

The rest of this paper is organized as follows. Section 2 provides a shot survey of some related works for object tracking. Section 3 concentrates on the proposed method, which utilizes a TLD scheme. Prototype system is implemented and experimental results for evaluating the proposed method are shown in Section 4. Finally, Section 5 presents our conclusion and future work.

2 RELATED WORKS

RECENTLY, many researches have been developed in the field of object tracking (Alper, 2006). Typically, visual tracking methods can be divided into two categories; generative approach and discriminative approach (Xiaoyu, 2016). Generative approach utilizes a model to describe the apparent characteristics, and minimizes the reconstruction error to search the desired object. On the contrary, discriminative approach supports a method to distinguish between object and the background. These algorithms are more robust, and currently become more popular in the field of visual tracking. Discriminative method is referred to as tracking-by-detecting, and deep learning belongs to this category.

In object recognition and tracking with video stream, one of the most important things is an accurate extraction of feature. The extracted feature descriptions are composed of feature descriptor and processing methods to extract and match between the query and the target. Good descriptor should have a robust ability to handle intensity, rotation, scale, and affine variations. This means that the features should be distinct, should be scale and rotation invariant, should not affect the viewpoint change, and should not take much time to extract. In object recognition, many researches and applications use SIFT (Scale Invariant Feature Transform) and SURF (Speeded-Up Robust Features) as algorithm for feature extraction (Yong, 2015). SIFT aims to resolve the practical problems in low level feature extraction and their use in matching images. SIFT involves two stages: feature extraction and description (David Lowe, 2004). The description step concerns use of the low-level features in object matching. Low level feature extraction within SIFT approach selects salient features in a manner invariant to image scale, rotation and partial invariance to change by illumination. However, as increase an image size using in SIFT algorithm, it will be plenty to calculate the amount of data and this leads a computing time as increasing exponentially, because of its high dimension characteristics (Martin, 2010). Using SURF, features are located using an approximation to the determinant of Hessian matrix (Herbert, 2008). For detail, we represent a SURF approach in the following sub-section. Anyway, SURF has fast feature extraction and feature descriptor to reduce complexity of the operation in the process of feature extraction and matching, compared to SIFT method (Jin, 2010). It leads good results and high speed by decreasing processing time through more efficient extracting method of the feature and descriptors.

Many tracking algorithms have been proposed in earlier researches. In tracking video sequence, an object is defined as anything which is interesting for analysis (Kaiqi, 2008). The aim of object tracking is to generate the trajectory of an object over time by locating its position in every frame of the input video. Object tracking approaches commonly use MeanShift and CamShift (Zhiyu, 2014). MeanShift is a kind of tracking algorithm based on external features, with which real-time tracking for non-rigid object (Ido, 2010). This algorithm is an efficient approach to tracking objects whose appearance is defined by histograms. In the process, key points in n-dimensional feature space as empirical probability density function, where dense regions in the feature space correspond to the local maxima or modes of the underlying distribution (Werner, 2010). CamShift is one of the most important algorithms for object tracking (Alexandre, 2004), which is an adaptation of the MeanShift approach in computer vision. A primary difference between CamShift and MeanShift algorithm is that MeanShift is based on static distributions, which are not updated unless the target experiences significant change in shape, while

CamShift uses continuously adaptive probability distributions. David proposed a low-cost extension to CamShift to resolve a problem that is not robust in complex backgrounds (David, 2010).

Deep learning technique, included in discriminative approach, recently provides impressive performance in computer vision and video tracking by huge margin. As well as, most of all good algorithms in scene classification, object detection and image analysis are based on deep learning (Xiaoyu, 2016). One of the first successful methods in this family is based on convolutional neural networks (Linnan, 2017). Jin presented deep neural network models to conduct long-term single object tracking with radial basis function classifier (Jonghoon, 2013). Ka proposed a visual object tracking using hierarchies of convolutional layers as non-linear of image pyramid and adaptive learn correlation filters on each convolutional layer (Chao, 2015). Bae proposed a robust online multiple object tracking method using confidence-based data association and discriminative deep appearance learning to handle track fragments and similar appearances of objects (Seung, 2018). Wu presented a simultaneous tracking, learning and parsing method, named AOGTracker, for unknown object tracking in video sequences with hierarchical and compositional And-Or graph representation (Tianfu, 2017). The main difference between deep learning approach and other approaches is that the feature representation is learned instead of being designed by the user (Jia, 2016), but with the drawback that many training samples is required for training the classifier. The key difference of this paper from other approach is that we provide a credibility level for the use of mobile environment. This method leads an advantage of reducing the amount of learning and increasing the adaptability in the mobile platform

3 PROPOSED METHOD

THIS section describes each component of the proposed scheme. In overview, the proposed method for visual object detecting and tracking has a simple cyclic architecture, which is based on set of inter-connected independent three modules. Each one deals with specific type of input, which is elaborated to provide relevant information to the connected module. As shown in Figure 1, the proposed system is initialized by feeding input video. First, *detecting module* extracts and identifies a desired object, then sends the detecting information and object information to learning and tracking module, respectively. Then, *tracking module* tracks the interest object using feature points, and passes trajectory and information of the computed credit level to check whether the desired object is traced well or not. *Learning module* gets the detecting data and credit values from detecting module and tracking module, respectively. This module finds interest object again with validity information when the desired object is missed or failed in the tracking, then pass them to detecting module to track continuously.

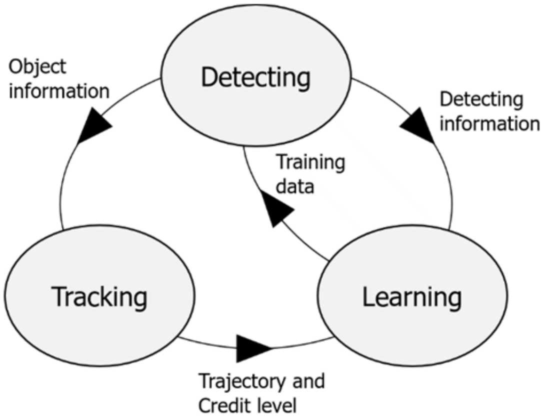

Figure 1. Diagram of the proposed cyclic architecture with interconnected three modules.

Object detection and tracking are described in Section 3.1 and Section 3.2, and learning focused on robust tracking with evaluation of validity is discussed in Section 3.3.

3.1 Object Detecting

Object detecting is to find and to localize the interest objects in an input stream. In this paper, the definition of *object* has various meanings. This means a single instance or an all class of objects, which is interest in the video stream by user. Object detection method is generally based on the local features of input image (or one of the frames from input video) (David, 2004) or a sliding window (Paul, 2001). The sliding window-based method scans the input image by a window of various sizes and for each window decide whether the underlying patch contains the interest object or not. Since this approach evaluates in every frame causing high complexity of computation, this paper focuses on local image features, and feature-based approach is commonly followed by three steps; (1) feature extraction, (2) feature recognition, and (3) model fitting (Tawfiq, 2016).

First, we utilize a learned database, constructed by feature point extraction and descriptor using SURF, to get information for the interested object and to detect its pose from input video sequences. Then, to rapidly retrieve and recognize the object information from the learned database for matching features of the current frame, we use a LSH (Locality Sensitive Hashing) algorithm. In the process of feature extraction, we use SURF algorithm to extract feature points and to detect and location the desired object with Fast-Hessian detector. Detection of the interested object is conceptually based on scale space theory. SURF uses an integral image as the determinant of Hessian (Li, 2014), which is formed by the origin with Equation (1).

$$\mathrm{I(X)} = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j) \qquad (1)$$

The integral image is utilized in Hessian matrix approximation to reduce the computing time effectively. The Hessian matrix $\mathrm{H}(X,\sigma)$ in X at scale σ is defined in Equation (2).

$$\mathrm{H(X,\sigma)} = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix} \qquad (2)$$

where $L_{xx}(X,\sigma)$ is a convolution of the second order of Gaussian filter $\frac{\partial^2}{\partial x^2} g(\sigma)$ with the image I at point X. $g(\sigma)$ is computed by Equation (3).

$$\mathrm{g(\sigma)} = \frac{1}{2\pi\sigma^2}\, e^{\frac{-(x^2+y^2)}{2\sigma^2}} \qquad (3)$$

Similarly, $L_{xy}(X,\sigma)$ and $L_{yy}(X,\sigma)$ is convolution of the filter for *xy* direction (diagonal) and *y* direction (vertical), respectively. To reduce the computing time, set of 9 × 9 box filter is used for approximations for Gaussian second order derivatives with $\sigma = 1.2$ and this value represent the lowest scale. We denote the Hessian determinant by D_{xx}, $D_{xy}$and D_{yy}. The weights applied to the rectangular regions are simply kept for computational efficiency by Equation (4).

$$\mathrm{Det}\left(H_{approx}\right) = D_{xx}\cdot D_{yy} - \left(0.9\times D_{xy}\right)^2 \qquad (4)$$

As scale space representation is implemented with image pyramid, scale space is a continuous function which is used to find the maximum values all possible scales (Christopher, 2009). Scale space in SURF is analyzed by up-scaling the filter size, instead of iteratively reducing the image size. Then it is divided into several octaves, which refer to a series of response results from convolving the same input image with the different sized filter that is increased. Thus, we divide the image into octaves. Each octave is sub-divided into constant number of scale level, and contains different scale image templates. We apply as image scale $\mathrm{s} = 1.2\times\frac{N}{9}$, with box filter scale N × N.

After obtained the approximation of Hessian matrix determinant in each layer, then maximum in the neighborhoods is interpolated within the image space to localize the interest points over the scales of image. At this time, we get a set of the interest points that has minimum strength determined by threshold value, as well as maximum/minimum in the scale space. The responses are then partitioned into 4 × 4 sub-windows, and this provides the results in 16 sets of *dx* and *dy* values.

For each sub-window to generate a set of entries of the feature vectors, we compute sum of values and magnitudes for both *dx* and *dy* with Equation (5).

$$\mathrm{V} = \left[\sum dx, \sum dy, \sum|dx|, \sum|dy|\right] \qquad (5)$$

where *dx* and *dy* are the horizontal and the vertical wavelet responses over each sub-window region, and $|dx|$ and $|dy|$ are sum of the polarity of the image intensity changes.

Next step is to retrieve an object information for matching with the features of the current frame from the learned database. In the matching techniques, KD (*k*-Dimension) tree is commonly used for nearest neighbor query, which is a binary space partitioning that recursively segments the feature space in the dimension with highest variance (Minjie, 2010). KD tree is useful data structure for applications (Rina, 2008). However, as growing the size of feature vectors, performance of KD tree is rapidly going down (Deepika, 2014). For fast matching of high dimensional features between the current frame and the database, we apply the locality sensitive hashing (LSH) to solve the problem of the performance in high dimensional nearest neighbor (Subhashree, 2016).

Basic concept of the LSH is that if point *p* and point *q* are close to each other, then there is a high probability they collision, but two points *p* and *q* are far from each other, they have a smaller probability. LSH function is required to satisfy two necessary conditions as following.

If $\mathrm{d(q,v)} \le d1$, then probability $\mathrm{h(q)} = \mathrm{h(v)}$ is at least p1

If $\mathrm{d(q,v)} > d2$, then probability $\mathrm{h(q)} = \mathrm{h(v)}$ is at most p2

where $\mathrm{d(q,v)}$ is a distance between *q* and *v*, *h*(*q*) and *h*(*v*) denote hashing value of *q* and *v*. If hashing function satisfies above conditions, a vector value $(d1, d2, p1, p2)$ is called as sensitive (Wei, 2015). As one or more hash functions of value $(d1, d2, p1, p2)$ are sensitive, we make one or more hash functions as LSH with the following steps.

(1) Creation of LSH;

(2) Querying model of LSH;

(3) Determination of parameter *K* and *L*;

According to the principal of LSH, the number of hash keys *K* and the number of hash tables *L* play an important role in the performance (Malcolm, 2008). To have a good performance of the LSH on matching query, we must carefully choose values of appropriate *K* and *L*, which ensures the following condition under a constant probability.

$$\text{if exist } v^* \in B(q,r_1), then\ g_j(v^*) = g_j(q) \qquad (6)$$

Total number of collisions with *q* is less than 3*L*, as Equation (7).

$$\sum_{j=1}^{L}\left|\left(P - B(q,r_2)\right)\cap\left(g_j^{-1}\left(g_j(q)\right)\right)\right| \le 3L \qquad (7)$$

3.2 Object Tracking

Object tracking module is to estimate and to trace a motion of the desired objects. Object tracker is typically under the assumption that the interest object is visible on all the computing sequences. Several type of representations for the object are used in the fields of academic and industrial (Alper, 2006). For example, there are points-based (David Servy, 2004), primitive geometric shapes-based (Ruwen, 2007), silhouette-based (Ajoy, 2014), articulated shape model-based, and skeletal model-based (Sumanth, 2007). This paper focuses on the object, represented by geometric shapes and their motion which are estimated between consecutive frames. Object tracking measures the poses of the desired object and its information using CamShift and optical flow. The pose of each object is evaluated and confirmed from base object information. The base object information is comprised with feature points, descriptor, window-included object, pose of the expected object, and its histogram, which are passed from the previous detecting module. In this step, the object is tracked by measuring of the feature points and its histogram, each method estimates the pose of the object and the expected window containing the object. We split the window into sub-windows, and then we extract a differentiation of the object information.

The tracked object from the sub-window within the current frame is estimated by measuring of the validity for learning of the base object information. To evaluate the validity of the object, we compare the information of the tracked object to the initial information, which is consisted of two fields; feature points and its histogram. In this work, we call the base object information as *Level*-1 information.

To make a level-1 information of the tracked object to compare with the base object information, we firstly define sub-windows, and then analysis morphological definition of the object, and make labels of the object with feature points. These steps are formally listed as the following.

(1) Defining sub-windows;

(2) Defining morphology and analyzing the object;

(3) Labelling feature points of the object;

We split a camera view into sub-windows, based on CamShift and optical flow, by choosing the window area of the expected object and by comparing the histogram of the sub-window with the base object histogram. Each sub-window is divided into positive area and negative area, according to the highest ratio of its histogram. We update the tracked object information with the information of the positive sub-windows that include matching key-points in the frame. Figure 2 shows sub-windows that contain the highest ratio of the object histogram.

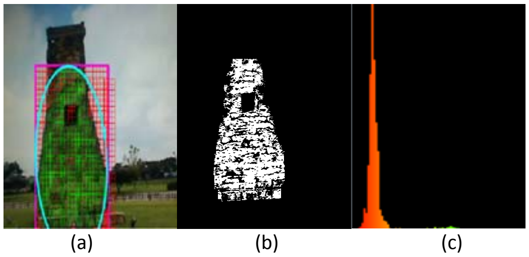

Figure 2. **Positive sub-window is shown in green, which is more similar with the color of object in (a). Negative sub-window is presented with red. The object mask is created by positive sub-window, as shown in (b), and its histogram of the adapted object is shown in (c).**

$$\begin{gathered} Label_0\left(k_1 \ni X_n(x_1, x_2 \dots, x_n)\right) \\ \vdots \\ Label_j\left(k_{j+1} \ni X_{j+1}(x_1, x_2 \dots, x_{j+1})\right) \end{gathered} \qquad (8)$$

The higher number of matching feature points between base object of *Level*-0 and the tracked object, the higher the probability of matching pixels. Thus, higher score of the validity level can be obtained with higher matching count. As the result of the module, the morphological characteristics map of the tracked object is shown in Figure 3.

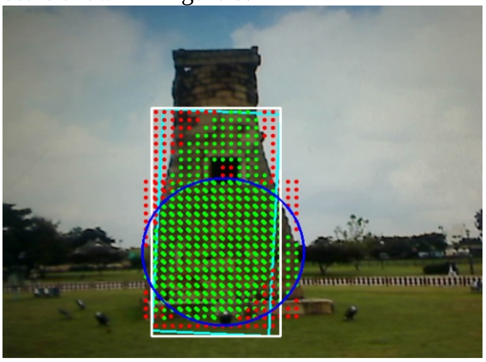

Figure 3. **Morphological characteristics map of the tracked object. Green shows the largest frequency of the object histogram, and red is extraneous to the object color.**

In this process, we need to consider a missing of the tracked object. When the tracking module loses the tracked object, we must re-detect an expected object by using the feature points and its descriptors with matching current frame to the trained feature vectors. From the input video stream, a change of the object poses (and/or occlusion of the view) can make the tracked object to miss from the tracking or to lose the object. Thus, we need to calculate a homography continuously to the object for successfully tracking.

Given a homography matrix *H* as Equation (9), this paper defines a discrimination method with a threshold for determining whether the tracking is still going correct or not, by Equation (10).

$$\mathrm{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \tag{9}$$

$$\angle \mathrm{A} \;\&\; \angle C \;\&\; \angle B \;\&\; \angle D < 180,\ \mathrm{D} = (\mathrm{h}_{11}\mathrm{h}_{22} - \mathrm{h}_{12}\mathrm{h}_{21}) \tag{10}$$

$$\mathrm{P} = \sqrt{h_{31}^2 + h_{32}^2}\,,\ \mathrm{sx} = \sqrt{h_{11}^2 + h_{21}^2}\,,$$
$$\mathrm{sy} = \sqrt{h_{12}^2 + h_{22}^2}$$

3.3 *Object Learning*

A learning method is employed in both detecting approach and tracking approach. Tracker use the learning module to adapt to changes of the object appearance, while detector use it to build better detecting model that covers various appearances of the object. Object detector is typically trained with assuming that all training data are labeled. It is too heavy to apply this assumption into our method, because we want to train an object detector on real-time processing by a single value to calculate a validity of tracking from a video stream. This leads us to make a formulation of validity level with self-learning method (Xiaojin, 2009).

The validity evaluation for object learning is a key proposed algorithm in this paper, which estimates the extracted objection information, and determines a validity level with comparing between the extraction object and the most reliable object, which is evaluated to the validity of the object. To estimate the validity level of the tracked object, we need to consider the object information, which are consisted of the positive sub-windows, the matched distribution of labels, the morphological shape of the object, and the expected windows. Those elements are based on CamShift and optical flow. We calculate the validity level using Equation (11).

$$Label_{n-1} = \begin{bmatrix} \frac{number\ of\ Labels}{Total\ Labels} \times \frac{number\ of\ frame}{Total\ Frames} \times \frac{matching\ X_n}{X_k}, \\ \left(0 \le H\left(Prob_{level_{k-1}} = (C_1 \cdots C_k),\right) \le 1\right) \end{bmatrix} \tag{11}$$

The validity of the tracked object is estimated, and each object is evaluated up to *Level*-5 based on the object which has the validity of *Level*-0. Thus, each object in the tracking is consisted of total 6 levels of validity. When the tracked object is missing to track down, detection module tries to detect the missing object again using the learned information of the object, and re-assigns the validity level using the object information within the current frame. The highest level of the validity by information of the re-learned object is re-assigned to the base object information again. Re-detection module includes a recognition process of the object and a matching process of sub-windows to the feature points to provide new depth of the validity level.

4 EXPERIMENTAL RESULTS

WE give a performance comparison to existing methods and analyze whether our method is suitable for detecting and tracking of visual objects on mobile environments. The proposed system is implemented in iOS platform using Objective-C and C++ with Xcode, running MacOS Sierra (10.12.1), 4GHz Intel Core i7 and 32GB DDR3 memory. In the implementation of the proposed method, we utilize three sample images, as shown in Figure 4, which are trained offline by training image data. Experiments are focused on modelling of target object with the trained object, and this is concentrated on how well does the proposed method detect a target object, and how robust does it an appearance changes, scale changes, and covered object. Figure 5 shows an example screenshot of the execution results, running on the change of rotation and scale in the interesting object.

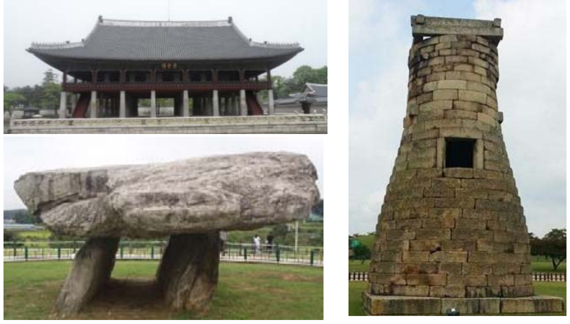

Figure 4. Sample training image set.

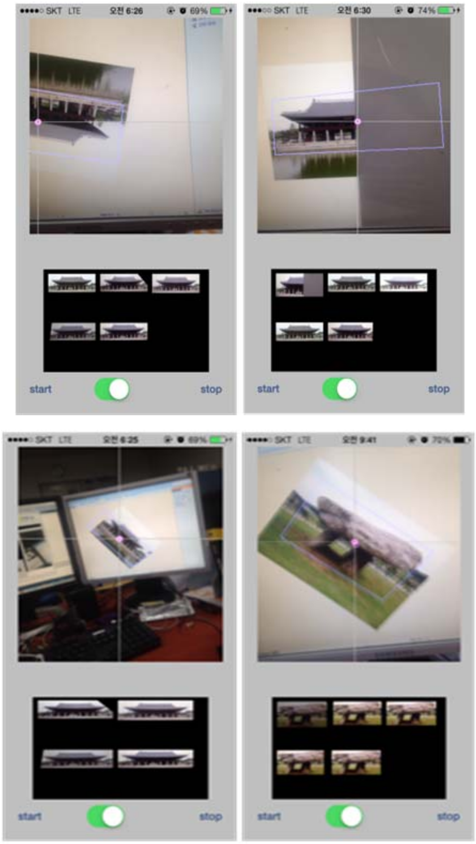

Figure 5. Screenshot of the tracking object over the change of rotation and scale.

Table 1 represents the comparison results of the object detection and tracking over the input video stream. Source codes of the compared algorithm in this experiment are downloaded from author's website (Cor, 2001) (Jianhong, 2011), and the results of the experiments for comparing the performance are computed by real-time running on mobile phone. To compare, we compute many successfully detected and tracked frames, which calculates a ratio of frames among the entire sequences that satisfies the following condition. The estimated tracking box where sufficiently overlapped with a ground truth bounding box is larger than 50%. The overlap precision how to compute here is defined as Equation (12) (Ines, 2006).

$$\text{Overlap Precision} = \frac{\#\ of\ frames\ that\ satisfies\ the\ condition}{\#\ of\ total\ frames} \times 100\ (\%) \qquad (12)$$

Table 1. Comparisons of the detecting and tracking efficiency using computing the overlap precision in real-time benchmark sequences.

Methods	Average overlap precision	Features extraction time
SURF+CamShift	84%	0.443
MIL	86%	0.721
Proposed method	89%	0.515

As the experimental results in Table 1, the proposed approach reveals more efficiency both on the overlap precision and feature extraction time, compared to the existing method such as Multiple Instance Learning (MIL) (Boris, 2011). However, the proposed method spends a little more time to extract the features with the efficient average overlap precision than SURF+ CamShift (Jianhong, 2011).

To compare the performance of the detecting and tracking to the existing researches, we use LTDT2014 datasets (Mario, 2014) and LOT datasets (Shaul, 2015) that are commonly used in the research. LTDT2014 datasets are collections of six video sequences, which provide an evaluation kit for comparison of tracking algorithms. They have several video sequences to experiment with scale changes and appearance changes. NissanSkylineChaseCropped (NSCC) sequence contains 3,742 frames, which has features that reflect a variation of the object size. On the contrary, Sitcom sequence contains 3,898 frames, and this dataset could be used to estimate the rotation of the object. These two sequences are used as a target experiment in mobile environment, which are good example of the experiments that reflect the characteristics of object movement. We use two sequences, Lemming and Skating, within LOT dataset for the experiments. While Lemming is used for object tracking with relatively lower movement of the target object, Skating is for experimental purposes to track the rapid appearance and disappearance of the object. Figure 6 shows sample videos for testing dataset, showing the ability of the detecting and tracking to cope with difficult states to trace, such as appeared new object, disappeared the tracked object by others covering scene, something suddenly changed by the background scene, and changed on the viewing scale.

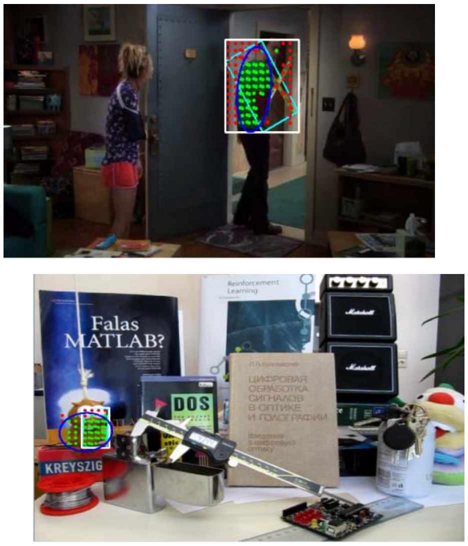

Figure 6. Screenshot of the tracking object using the proposed method with Sitcom and Lemming data.

Table 2 and Table 3 show comparisons of the proposed tracker with the existing other tracking approaches, as Iterative Visual Tracking (IVT) (David, 2008), Online AdaBoost (OAB) (Helmut, 2006), Multiple Instance Learning (MIL) (Boris, 2011), Visual Tracking Decomposition (VTD) (Junseok, 2010), Tracking-Learning-Detection (TLD) (Zdenek, 2012), Sparse Collaborative Model (SCM) (Wei, 2014), Adaptive Structural Local Sparse Appearance Model (ASLSA) (Xu, 2012), and Locally Orderless Tracking (LOT) (Shaul, 2015). The proposed tracking algorithm achieves more improved results with Lemming and NSCC sequences, and outperforms others in the environment of similar background and severe scale changes. The severe appearance of object changes, such as Skating and Sitcom, reveal the lowest overlap precision. However, the proposed approach shows robustness in difficult status, such as appearance object changes, scale changes, disappeared object covered other objects, and similar background. Especially, the proposed scheme is more robust in the case of the changes via rotation and view scale, since we believe that we applied the proposed algorithm with estimation of validity level for the tracked object and homography in the estimated window.

Table 2. Performance of the ratio of frames for which the PASCAL criterion was $a_0 > 0.5$

Sequence	Lemming	Skating
Frames	1,336	707
IVT	16.2	3.8
OAB	37.1	8.8
MIL	37.6	9.8
VTD	54.3	11.5
TLD	25.2	4.1
SCM	16.6	11.9
ASLSA	16.8	5.1
LOT	73.8	29.4
Proposed method	87.6	31.7

Table 3. Comparison of the performance using overlap precision with Matrioska tracker in LTDT sequences.

Sequence	Sitcom	NSCC
Frames	3,898	3,742
Matrioska overlap	48.6	85.4
Proposed method overlap	51.3	88.2

The experimental results reveal that the proposed method produced a significant improvement in real-time environment. Especially, the system is robust to scale changes and low movement of the target object, as reported in the result with NSCC and Lemming. This means that the proposed method has a strength in mobile platform where zooming function is relatively weak in the mobile devices. However, the system may have a weakness in the rotation due to hand shake in the use of smartphone. Additionally, another checkpoint is remained into computational complexity and memory efficiency. Mobile device usually spends many computing powers to track the desired object with constraint resources. Because the proposed method requires more computing time than SURF+CamShift by the result, enhancement of time is necessary for computing power, and this is remained in future work.

5 CONCLUSION

OBJECT tracking algorithm plays a critical task in many application of computer vision. Typical examples of this kind of applications are automated video analysis, video surveillance and traffic monitoring system. In this paper, we proposed an efficient object detection and tracking approach, which is a robust method on tracking of an object in a video sequence, where the object changes appearance and its poses moving in and out of the camera view. The proposed scheme decomposes the tasks into three modules; detecting, tracking and learning. This paper has a focus on the learning component, which generates a validity level of the tracking object to evaluate the object whether tracking goes on correctly or not. A real-time implementation of the proposed system is described in detail, and experiments are performed to measure the performance of the visual object tracking. Superiority of the proposed approach with respect to the competitors is clearly demonstrated, and the experimental results reveal that our work is more robust than the traditional methods in the case of the object changes.

For future works, we have a plan to test performance of the proposed system with larger scale used in tracking researches and common real-life video sequences like YouTube dataset, and we are going to reinforce the performance evaluation of each module to compare with the state-of-the-art approaches. Then we extend our work to incorporate other attributes, such as gyroscope sensor and GPS sensor. We are going to complete the development of deployable application that applies to augmented reality of heritage and cultural contents, and to video surveillance system with recognizing multiple objects.

6 REFERENCES

T. A. Al-asadi and A. J. Obaid, (2016). Object Detection and Recognition by using enhanced Speeded-Up Robust Feature, *International Journal of Computer Science and Network Security*, 16(4), 66-71.

B. Babenko, M.-H. Yang, and S. Belongie, (2011). Robust Object Tracking with Online Multiple Instance Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8), 1619-1632.

S.-H. Bae and K.-J. Yoon, (2018). Confidence-based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3), 595-610.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, (2008). SURF: Speeded-Up Robust Features, *Computer Vision and Image Understanding*, 110(3), 346-359.

E. Carniani, G. Costantino, F. Marino, F. Martinelli, and P. Mori, (2016). Enhancing Video Surveillance with Usage Control and Privacy-Preserving Solutions, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 7(4), 20-40.

W. Cen and K. Miao, (2015). An Improved Algorithm for Locality-Sensitive Hashing, *International Conference on Computer Science and Education*, 61-64.

D. Chen, F. Bai, P. Li, and T. Wang, (2009). BEST: A Real-time Tracking Method for Scout Robot, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1001-1006.

I. Cherif, V. Solachidis, and I. Pitas, (2006). A Tracking Framework for Accurate Face Localization, *International Federation for Information Processing*, 385-393.

D. Comaniciu, V. Ramesh, and P. Meer, (2003). Kernel-based Object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5), 564-577.

C. Evans, (2009). Notes on the Open SURF Library, *University of Bristol*.

D. Exner, E. Bruns, D. Kurz, and A. Grundhofer, (2010). Fast and Robust CamShift Tracking, *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

X. Feng, W. Mei, and D. Su, (2016). A Review of Visual Tracking with Deep Learning, *International Conference on Artificial Intelligence and Industrial Engineering*, 231-234.

A. R.J. Francois, (2004). CAMSHIFT Tracker Design Experiments with Intel OpenCV and SAI, *International Conference on Pattern Recognition*, 4, 1-11.

H. Grabner, M. Garbner, and H. Bischof, (2006). Real-Time Tracking via On-line Boosting, *Proceedings British Machine Vision Conference*, 1, 47-56.

K. Huang, L. Wang, T. Tan, and S. Maybank, (2008). A Real-time Object Detecting and Tracking System for Outdoor Night Surveillance, *Pattern Recognition*, 41(1), 432-444.

X. Jia, H. Lu and M.-H. Yang, (2012). Visual Tracking via Adaptive Structural Local Sparse Appearance Model, *IEEE Conference on Computer Vision and Pattern Recognition*, 1822-1829.

J. Jin, A. Dundar, J. Bates, C. Farabet, and E. Culurciello, (2013). Tracking with Deep Neural Networks, *Conference on Information Sciences and Systems*, 1-5.

S. K. R. Kaditham and A. R. Pais, (2007). Model based Tracking, *International Conference on Intelligent Computing*, 1018-1025.

Z. Kalal, K. Mikolajczyk and J. Matas, (2012). Tracking-Learning-Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7), 1409-1422.

W. Kloihofer and M. Kampel, (2010). Interest Point based Tracking, *International Conference on Pattern Recognition*, 3549-3552.

J. Kwon and K. M. Lee, (2010). Visual Tracking Decomposition, *IEEE Conference on Computer Vision and Pattern Recognition*, 1269-1276.

Y.-H. Lee and H.-J. Kim, (2015). Evaluation of Feature Extraction and Matching Algorithms for the use of Mobile Application, Journal of the Semiconductor and Display Technology, 14(4), 56-60.

I. Leichter, M. Lindenbaum, and E. Rivlin, (2010). Mean Shift tracking with multiple reference color histograms, *Computer Vision and Image Understanding*, 114(3), 400-408.

J. Li, J. Zhang, Z. Zhou, W. Guo, B. Wang, and Q. Zhao, (2011). Object Tracking using Improved Camshift with SURF Method, International Workshop on Open-Source Software for Scientific Computation,

M. Li, L. Wang, and Y. Hao, (2010). Image Matching based on SIFT features and KD-Tree, *International Conference on Computer Engineering and Technology*, 4, 218-222.

L. Li, (2014). Image Matching Algorithm based on Feature Point and DAISY Descriptor, *Journal of Multimedia*, 9(6), 829-834.

X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick and A. V. Hengel, (2013). A Survey of Appearance Models in Visual Object Tracking, *ACM Transactions on Intelligent Systems and Technology*, 4(4), 1-58.

D. G. Lowe, (2004). Distinctive Image Features from Scale-Invariant Keypoints, *International Journal of Computer Vision*, 60(2), 91-110.

C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, (2015). Hierarchical Convolutional Features for Visual Tracking, *International Conference on Computer Vision*, 3074-3082.

M. E. Maresca and A. Petrosino, (2014). The Matrioska Tracking Algorithm on LTDT2014 Dataset, *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 720-725.

A. Mondal, S. Ghosh and A. Ghosh, (2014). Efficient Silhouette-based Contour Tracking using Local Information, *Soft Computing*, 20(2), 785-805.

K. Nishida, T. Kurita, Y. Ogiuchi, and M. Higashikubo, (2010). Visual Tracking Algorithm using Pixel-Pair Feature, *International Conference on Pattern Recognition*, 1808-1811.

S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, (2015). Locally Orderless Tracking, *International Journal of Computer Vision*, 111(2), 213-228.

R. Panigrahy, (2008). An Improved Algorithm Finding Nearest Neighbor using KD-trees, *Latin American Symposium on Theoretical Informatics LATIN*, 387-398.

S. P. Patil, (2016). Techniques and Methods for Detection and Tracking of Moving Object in a Video, *International Journal of Innovative Research in Computer and Communication Engineering*, 4(5), 8116-8121.

D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, (2008). Incremental Learning for Robust Visual Tracking, *International Journal of Computer Vision*, 77(1-3), 125-141.

R. Schnabel, R. Wahl, and R. Klein, (2007). Efficient RANSAC for Point-Cloud Shape Detection, *Computer Graphics Forum, Wiley Online Library*, 26, 214-226.

D. Servy, E.-K.-Meier, and L. Van Gool, (2004). Probabilistic Object Tracking using Multiple Features, *IEEE International Conference of Pattern Recognition*, 1-4.

J. Shi and C. Tomasi, (1994). Good Features to Track, *IEEE Conference on Computer Vision and Pattern Recognition*, 593-600.

M. Slaney and M. Casey, (2008). Locality-Sensitive Hashing for Finding Nearest Neighbors, *IEEE Signal Processing Magazine*, 25(2), 128-131.

M. Stommel, (2010). Binarising SIFT-Descriptors to Reduce the Curse of Dimensionality in Histogram-Based Object Recognition, *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 3(1), 25-36.

V.K. Subhashree and C. Tharini, (2016). Real-Time Implementation of Locality Sensitive Hashing using NI WSN and LabVIEW for Outlier Detection in Wireless Sensor Networks, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 7(3), 22-39.

C. J. Veeman, M.J.T. Reingers, and E. Backer, (2001). Resolving Motion Correspondence for Densely Moving Points, *IEEE Transactions on Pattern Analysis*, 23(1), 54-72.

D. Verma, N. Kakkar, and N. Mehan, (2014). Comparison of Brute-Force and K-D Tree Algorithm, *International Journal of Advanced Research in Computer and Communication Engineering*, 3(1), 5291-5297.

P. Viola and M. Jones, (2001). Rapid Object Detection using a Boosted Cascade of Simple Features, *Conference on Computer Vision and Pattern Recognition*, 511-518.

J. Wang, F. He, X. Zhang, and Y. Gao, (2010). Tracking Objects through Occlusions Using Improved Kalman Filter, *International Conference on Advanced Computer Control*, 223-228.

T. Wu, Y. Lu, and S.-C. Zhu, (2017). Online Object Tracking, Learning and Parsing with And-Or Graphs, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12), 2465-2480.

H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, (2011). Recent Advances and Trends in Visual Tracking: A Review, *Neuro-computing*, 74(18), 3823-3831.

A. Yilmaz, O. Javed, and M. Shah, (2006). Object Tracking: A Survey, *ACM Computing Surveys*, 38(4), 1-45.

J. Zhang, L. Yang, and X. Wu, (2016). A Survey on Visual Tracking via Convolutional Neural Networks, *International Conference on Computer and Communications*, 474-479.

W. Zhong, H. Lu, and M. H. Yang, (2014). Robust Object Tracking via Sparse Collaborative Appearance Model, *IEEE Transactions on Image Processing*, 23(5), 2356-2368.

Z. Zhou, D. Wu, X. Peng, Z. Zhu and K. Luo, (2014). Object Tracking Based on CamShift with Multi-feature Fusion, *Journal of Software*, 9(1), 147-153.

L. Zhu, L. Yand, D. Zhang, and L. Zhang, (2017). Learning a Real-time Generic Tracker using Convolutional Neural Networks, *International Conference on Multimedia and Expo*, 1219-1224.

X. Zhu and A. B. Goldberg, (2009). Introduction to Semi-Supervised Learning, *Morgan & Claypool Publishers*.

7 ACKNOWLEDGEMENTS

THIS research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2015R1D1A1A01061152).

8 AUTHORS

Yong-Hwan Lee received the M.S. degree in Computer Science and Ph.D. degree in Electronics and Computer Engineering from Dankook University at Korea in February 1995 and February 2007, respectively. Currently, he is an assistant professor at Department of Digital Contents, Wonkwang University, Korea. His research areas include Image Retrieval, Computer Vision and Pattern Recognition, Augmented Reality, Mobile Programming and Multimedia Communication.

Hyochang Ahn received the M.S. and Ph.D. degrees in Electronics and Computer Engineering from Dankook University, Korea, in 2006 and 2012, respectively. He was a research professor at Dankook University, Korea, from 2014 to 2016. Currently, he is working as Far East University, Korea. Computer Vision, Embedded System and Mobile Programming.

Hyo-Beom Ahn received the B.S. in Computer Science and M.S., and Ph.D. in Computer Science and Statistics from Dankook University, Korea in 1992, 1994 and 2002, respectively. Since then, he has been with the Department of Information and Telecommunication, Kongju National University, Korea. His main research interests include Computer Networks, Network Security, Smart Grid Security and Application and Industrial Control System Security.

Sun-Young Lee was born in Busan, Korea, in 1971. She received the B.S. and M.S. degrees in Computer Science from PuKyong National University, Busan, in 1995 and the Ph.D. degree in Information and Communication Engineering from the University of Tokyo, Japan, in 2001. From 2002 to 2003, she was a lecturer at Soonchunhyang University. Since 2004, she has been an Associate Professor with the Department of Information Security Engineering, Soonchunhyang University, Asan, Korea. Her research interests include Cryptography, Secure Protocol, Secure Healthcare System and Security Management.