Human Head Tracking Based on Inheritance and Evolution Concept

Yi Hu, Tetsuya Takamori
Fujifilm Corporation, Japan
798, Miyanodai, Kaisei-machi, Ashigarakami-gun, Kanagawa, 258-8538 JAPAN
{yi_hu, testuya_takamori}@fujifilm.co.jp

Abstract

This paper presents a method for tracking human head in cluttered scenes to achieve robustness to occlusions and environment change by introducing inheritance and evolution concept into tracking system. Different from most works on tracking where detection and tracking are loosely coupled, we view the essence of object detection and tracking as a process of modeling object class classifier (in detection stage) and object instance classifier (in tracking stage), they are coordinated in the class/instance relationship. At first we trained a head generic class classifier in off-line learning, which is a hierarchical structure where selected features are arranged from rough to detail and from low to high level. Then we derived the tracker's instance classifier from the base generic head classifier. The instance classifier is also a hierarchical structure. Its features in low and middle levels are initialized by inheriting low and rough features from generic base class classifier. Its high level features, the individual identification parts, are learned and evolved on-line with boosting method. By incorporating inheritance and evolution into tracking, the tracker not only adapts itself to the surrounding change, but also gains the ability naturally to distinguish it from other instances of the same class. The experiment shows its effectiveness.

1. Introduction and related work

Human detection and tracking plays an important role in intelligent surveillance monitoring, automatic customer information gathering and analyzing in shops and stores etc. Though a lot of research has been undergoing ranging from applications to noble algorithm, developing robust human detection and tracking algorithm is still challenge due to factors such as noisy input, illumination variation, cluttered backgrounds, occlusion, and human appearance change due to motion and articulation. Considering the characteristics of image sequences taken from surveillance camera at stores where the scene is in crowds and occlusion happens heavily, we started from doing human head detection and head tracking.

Most of works on human head detection and tracking model the human head as an ellipse shape [1] or a predefined template [2]. However, since various hairstyles and photographical angles of surveillance camera exist, there are head objects cannot be modeled as ellipse shape, which limits the head type that can be detected.

In many works on object tracking, the object classifier in a tracker is either directly a clone from the detector or is independently designed. Yuk et al. [2] applied the same head model for both detection and tracking. Their approach is difficult to deal with the occlusion-merge-split problem in tracking multiple people. On the other hand, Aviadan [3] and Grabner et al. [4] treat tracking as a classification problem and train a dedicated classifier for the tracker with ensemble learning method, which shows good abilities to deal with environment change in some degree. However, this kind of loosely coupling between detector and tracker, and weak learning because of fewer teacher examples limit its adaptability to severe environment.

Andriluka et al. [5] introduced a method combining the advantages of both detection and tracking in a single framework, where the combining is established based on temporal coherency among the results of detection and tracking. We address the same problem of tracking multiple people in complex real world scenes but in a different view.

The first contribution of this paper is that we coordinate detection and tracking in a class/instance relationship. We derived the tracker's instance classifier from detector's class classifier and evolved it through on-line ensemble learning. The second contribution is the extension of traditional planar sliding window structure (or object appearance structure) to a layered structure which accelerates detection speed and allows inheritance. The third contribution is a seamless connection between motion detection and object classification through frame difference in sliding window unit. The fourth contribution is a head (including face) detector with new feature types.

The rest of paper is organized as follow: Section 2 introduces the head generic class classifier. Section 3 first explains the relationship between detection and tracking, and then details the process to derive and evolve an instance classifier of a tracker. Section 4 shows the experiment. Section 5 draws the conclusion.

2. Training a head generic class classifier in off-line learning

As explained above, ellipse shape or predefined shape model limits the head type can be detected, we modeled the head appearance with a series of classifiers trained from head examples, which has a benefit that many human features such as eye, mouth, ear and head outline or shoulder line can be captured. Different from Viola and Jones [6] where the object model is in a plane structure, our head model is a cubic structure consisted of three layers in the reversed pyramid form shown by Fig.1 (a). These layers L_2 ~L_0 are arranged from low resolution to high resolution, where L_2 is 8 x 8 pixels, L_1 is 16 x 16 pixels, and L_0 is 32 x 32 pixels. We collect head samples

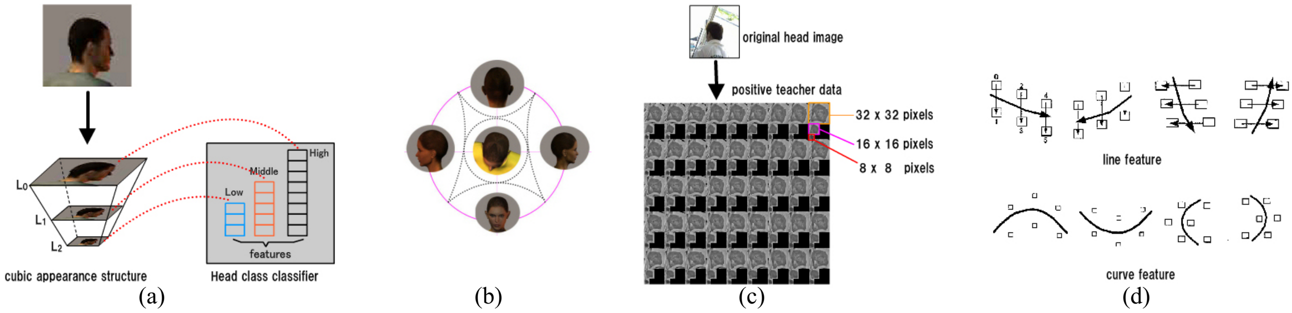


Fig.1 (a) cubic appearance structure; (b) 5 categories of head types; (c) one sample of teacher data; (d) line/curve feature types

and manually cluster them into 5 categories shown by Fig.1 (b). For each category, we trained a strong classifier just responding to this head view.

Each of the collected samples is firstly preprocessed with illumination corrected. We generate additional positive training data by artificially creating variations to the original positive training images. The purpose in doing so is to increase the accuracy of probability estimation and make the classifier more robust to head pose change. For each positive teacher image, we vary the size, the aspect ratio and the image orientation to generate 45 synthetic variations through small controlled variations in orientation, size and aspect ratio. Figure 1.(c) shows 45 varieties of a head teacher image and each variety is a 3-layered structure.

The training for each category is done in a hierarchical order. At first we use the teacher data of L_2 as training data to get the low/rough feature, then use the teacher data of L_1 to get the middle feature, finally use the teacher data of L_0 to acquire the high/detail features. We use the Haar-like features [6] and the features shown by Fig.1 (d) as weak classifiers. These line/curve features are features of triplet blocks which are arranged to sandwich a line or a curve. We apply these features to extract head characteristics on face /head contour and shoulder lines.

As a result the learned head classifier is a hierarchical structure from rough to detail feature cascaded by a series of weak classifiers. Each weak classifier is a distribution based on a single feature, and represented with a histogram with 256 bins. The benefits of this hierarchical classifier not only make it possible to inherit rough and low features, but also accelerate detection speed. In detection stage at each place where the sliding window locates, we first calculate the rough features and then, according to the result, to decide whether to go to high level features.

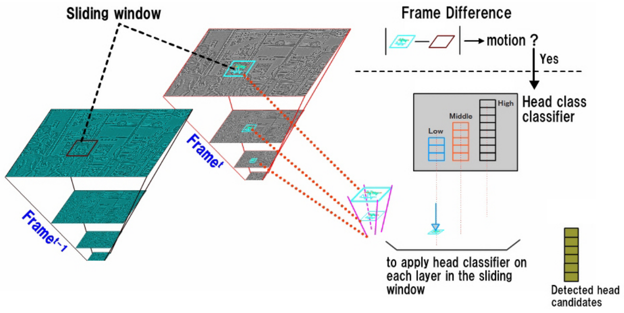


Fig. 2 A seamless connection from motion detection to classifier by doing frame difference in sliding window unit.

Motion detection is done in frame difference method. Different from the ordinary frame difference in pixel unit, we do the frame difference in sliding window unit as shown by Fig.2., where image integration [6] at L_0 of current frame t and previous frame $t-1$ is differenced to find motions. If there is no motion detected, the sliding window goes to next position. If a motion is found, the head classifier will be applied at this motion place. This kind of frame difference in sliding window unit acts as a seamless connection between motion detection and object classification. It greatly accelerates the detection speed and reduces false detection alarm.

3. Deriving and training an instance classifier in on-line learning

At first we examine the relationship between detection and tracking. Then we derive the instance classifier of a tracker from above base generic head classifier, as object-oriented programming does. After that we evolve the instance classifier so it is adaptable to the surroundings.

3.1 The relationship between detection and tracking

In a tracking system the detection and tracking are closely related with each other. Detection is a preprocessing process where the object is detected globally. If an object is found, it launches and initializes a tracker and passes the found object to the tracker. The tracker then starts tracking this object. Once the tracker fails in tracking, it asks the detector to do detection again. Their interface interaction is shown in Fig.3.

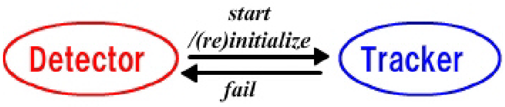


Fig.3 The interface interaction of detector and tracker.

If we examine detector and tracker much more, we will find both detector and tracker are doing “detection” job. The difference lies in the roles they are taking. Detector does the job globally and it uses static features to detect object in a class level. On the other hand, tracker does this job locally with the help of status estimation and it applies both static and dynamic features to find the individual object. They have a global/local, class/instance, static/dynamic relationship shown by Table 1. Here we constrain a tracker just to track a single object and name the classification part in this tracker as an instance classifier or individual classifier, which corresponds to the object being tracked.

Table 1. The relationship of detector and tracker.

	Detector	Tracker
search range	global	local
classifier	class	instance/class
feature	static	static/dynamic
history data	not used	used to do estimation

The instance classifier shows polymorphism in classification. For example, assuming a tracker is tracking a person called Mr. A, if there is no other person near surrounding, it behaves as a class classifier. But when another person Mr. B is walking nearer and crossing to Mr. A, then it has to distinguish Mr. A from Mr. B. In this case it behaves as an instance classifier.

Because of the above characteristics of instance classifier, especially its polymorphism property, we construct it from the detector's class classifier. This construction can be done through inheritance and evolution shown by Fig.4.

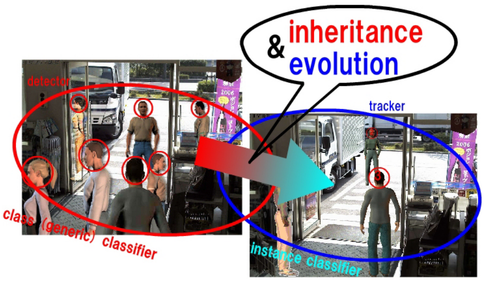


Fig.4 Inheritance and evolution act as a bridge between generic class classifier and instance classifier.

3.2 Deriving the instance classifier by inheritance

Based on the above head generic class classifier, the instance classifier inherits low and rough features of the parent classifier by duplicating them (both of features and the weak classifier's distribution). This is complete inheritance. For the middle and high level features, the instance classifier can choose and inherit part of them. This is shown in Fig.5. An instance classifier's polymorphism in "class form" is realized by inheritance. In our application we assume that both rough and middle features are 100% inherited, and the detail features are 60% inherited.

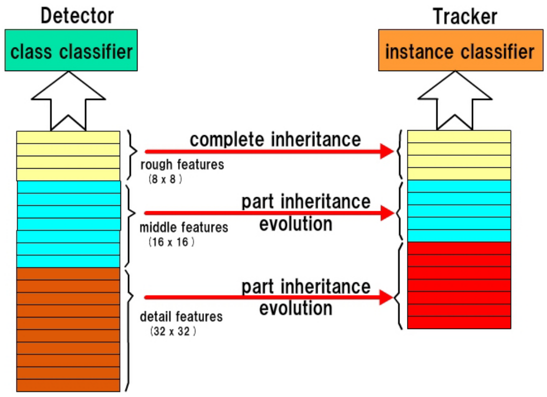


Fig. 5 Derive an instance classifier by inheritance.

3.3 Evolving the instance classifier

An instance classifier's polymorphism in "individual or instance form" is realized by evolution. We implement this evolution with on-line ensemble training. This is similar to the work of Aviadan [3] and Grabner et. al [4]. Our learning, however, is an evolution process shown by Fig.6. Assuming there are P features in the instance classifier, then the evolution process is as follows:

1. Select N top inferior features from the instance classifier by evaluating on current object status.
2. Remove the N features.
3. Do on-line learning on teacher data and select N features from feature pool.
4. Append the newly selected N features to the end of the instance classifier.
5. Update the instance classifier.

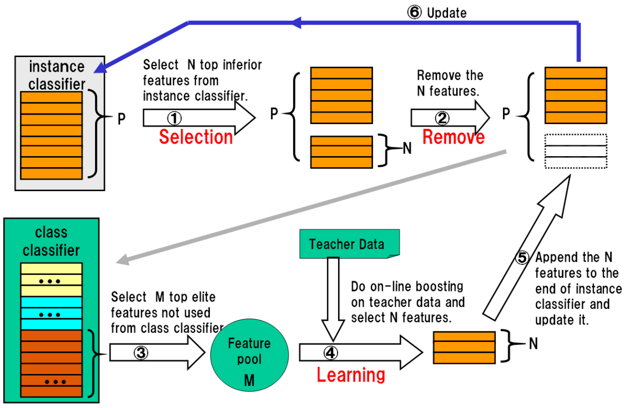

⑥ Update
instance classifier
P
Select N top inferior features from instance classifier.
① Selection
P
N
Remove the N features.
② Remove
P
class classifier
Select M top elite features not used from class classifier
③
Feature pool M
Teacher Data
Do on-line boosting on teacher data and select N features.
④ Learning
N
⑤ Append the N features to the end of instance classifier and update it.

Fig.6 Update an instance classifier dynamically by evolving.

3.4 Collecting teacher data on-line

The teacher data used in on-line training is exampled in Fig.7. The positive teacher data is the sub-image of the head being tracked in the current frame and in the previous frame. Considering the change of head appearance, the subimages centered in the head are also added into the positive teacher date set.

The negative teacher data consists of the following

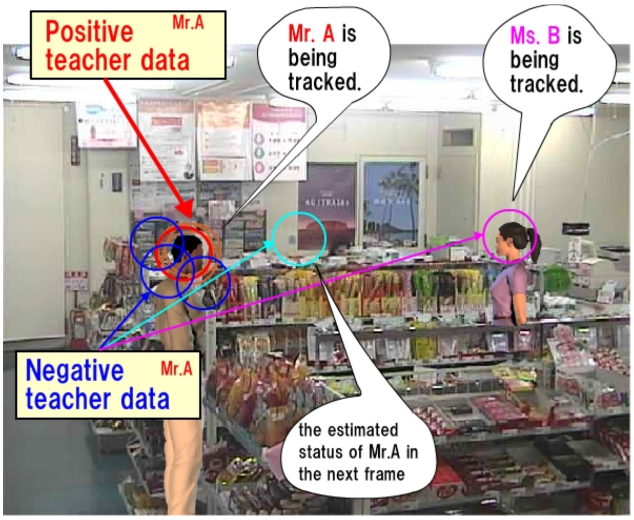

Positive Mr.A teacher data
Mr. A is being tracked.
Ms. B is being tracked.
Negative Mr.A teacher data
the estimated status of Mr.A in the next frame

Fig.7 The teacher data for on-line learning.

three types of subimages.

1. The subimages of surrounding background where the head is being tracked.
2. The subimages at which the head being tracked is estimated to exist at the next frame.
3. The subimages of other instances near the head being tracked.

By learning through the above negative teacher data, the instance classifier not only distinguish itself from the surrounding environment, but also distinguish itself from other instance object.

3.5. Controlling the learning degree

The on-line learning is not done at every frame. Instance classifier evolution will skip the following cases.

1. The head being tracked is at static status. Since there is no change in the appearance, we can skip the learning process.
2. The response of instance classifier is in a high positive value, which means our instance classifier already learned the status information (appearance of both head and environment) so it is not necessary to learn it again.
3. The response of instance classifier is lower than a given threshold value. This means that our instance classifier cannot be sure whether the currently estimated object status needs to be updated. To avoid the tracker going in wrong direction, the learning is skipped in this case.

4. Experiment and Results

The proposed approach was implemented and evaluated with surveillance image sequences of a convenience store. The tracking system is shown in Fig.8 where color information is not used. Nearly 85% human heads are correctly tracked in about 2000 frames with a frame rate of 5 on a machine of 3.0GHz. Fig.9 shows some example frames of tracking output. The test results show the proposed approach is effective.

5. Conclusion and future work

We have presented a tracking method where instance classifier's polymorphism is realized through inheritance and evolution. We begin by training a head generic class classifier detector in off-line learning method. Then we derive the instance classifier from the detector's class classifier and evolve it with on-line learning method. By incorporating inheritance and evolution, the instance classifier derived and evolved not only shows its adaptability to environment change, but also gains the ability naturally in distinguishing itself from other instances of the same class. The approach has been tested with real videos, and the results show its usefulness. Some remaining problems such as smooth transfer among instance classifiers of different views will be discussed in the near future.

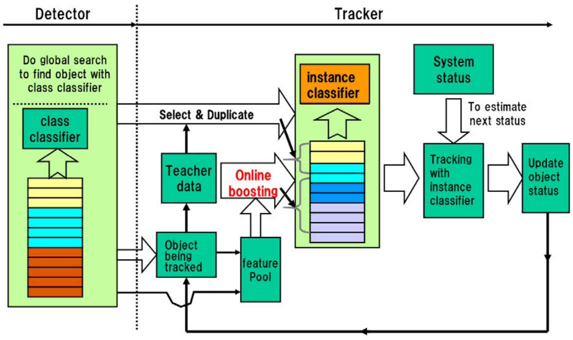


Fig.8 The outline of proposed tracking system.

References

[1] Alexander Barth and Rainer Herpers: "Robust Head Detection and Tracking in Cluttered Workshop Environments Using GMM", Pattern Recognition, vol.3663, pp.442-450, 2005.

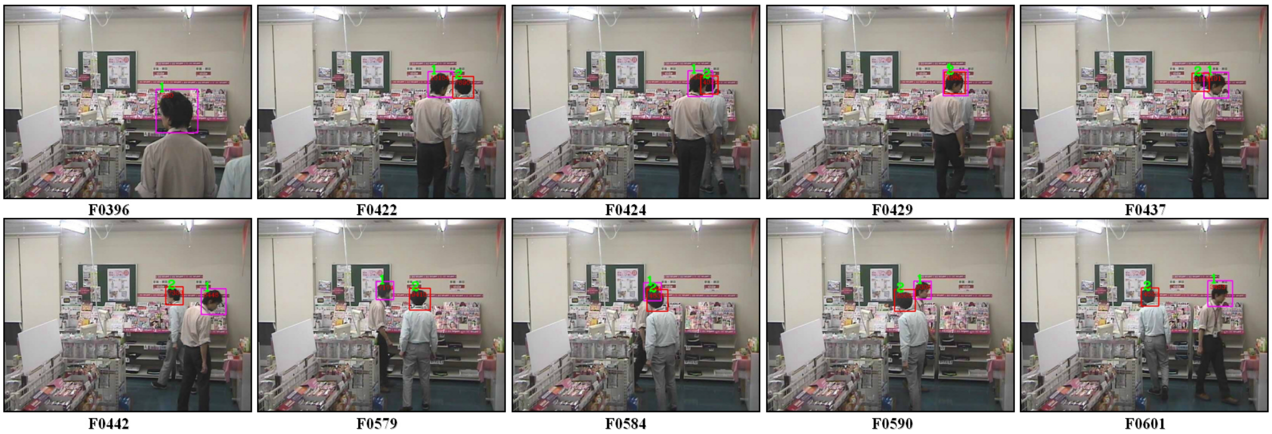
[2] Jacky S.C. Yuk et al. : "Real-time Multiple Head Shape Detection and Tracking System with Decentralized Trackers", Proceedings of the Sixth International Conference on Intelligent Systems Design and Application (ISDA'06), vol.02, pp.384-389, 2006.

[3] Shai Aviadan : "Ensemble Tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.29, issue.2, pp.261-271, Feb., 2007.

[4] Helmut Grabner, Michael Grabner: "Real-time Tracking via On-line Boosting", vol.1, pp.47-57, BMVC2006.

[5] Andriluka, M. Roth, S. Schiele, B.: "People-tracking-by- detection and people-detection-by-tracking", CVPR2008, pp.1-8, Jun., 2008.

[6] P. Viola and M. Jones: "Rapid object detection using a boosted cascade of simple features", in Proc. 2001 IEEE Conference on Computer Vision and Pattern Recognition, vol.1, pp.511-518, Dec., 2001.


F0396 F0422 F0424 F0429 F0437

F0442 F0579 F0584 F0590 F0601

Fig.9 The examples of tracking output with the proposed method on videos taken from surveillance camera.