INDEPENDENT SELECTION AND VALIDATION FOR TRACKING-LEARNING-DETECTION

Helena de Almeida Maia *Fábio Luiz Marinho de Oliveira* *Marcelo Bernardes Vieira*

Universidade Federal de Juiz de Fora, PGCC/DCC/ICE
Cidade Universitária, CEP: 36036-330, Juiz de Fora, MG, Brazil

ABSTRACT

On the problem of tracking objects in videos, a recent and distinguished approach combining tracking and detection methods is the TLD framework. The detector identifies the object by its supposedly confirmed appearances. The tracker inserts new appearances into the model using apparent motion. Their outcomes are integrated by using the same similarity metric of the detector which, in our point of view, leads to biased results. We propose a mediator method to integrate the motion tracker and detector by combining their estimations. Our results show that when the mediaton strategy is independent of both tracker/detector metrics, the overall tracking is improved for objects with high appearance variations throughout the video.

Index Terms— template tracking, semisupervised learning, tracking-learning-detection framework

1. INTRODUCTION

Object tracking is a fundamental task for several areas of research such as surveillance and augmented reality. Tracking provides an object position over time, so the system may be able to analyze the object behavior or to artificially produce new objects over it. The large variety of trackers in the literature mainly differ in the object modeling and how the trajectories are obtained. This work focuses on the template tracking problem which generally has two main approaches: tracking by motion model [1, 2, 3, 4] and by detection [5, 6]. Based on the premise that both approaches can be used at a time, we propose a *mediator*, composed by a selector and a validator, which decides whose tracker is more suitable for a given frame.

Tracking by motion model is based on motion estimation methods, mainly optical flow. It consists of optimized searches replacing the template at short intervals. Due to this updating process, the tracker is capable of adapting to new object appearances, but retaining only one template at a time. This becomes an issue when new appearances are not related to the original sample, caused by either gradual or abrupt insertion of background in the template. The error accumulation during tracking is known as the drift problem, which might cause the tracker to permanently lose the appearance of the original object, requiring re-initialization.

In tracking by detection, the object trajectory is obtained through independent detections at each frame. The detector is a classifier that examines the whole frame and decides about object presence or absence in each region. Generally, the detector is trained offline and requires a large set of samples and an exhaustive training or it gets outdated quickly. For this reason, many recent works propose semisupervised training that collects samples automatically at runtime [6, 7, 8, 9]. The key issue of semisupervised learning is the method for sample collection. It must accept new object appearances avoiding noise and other objects which may cover it. By using improper collected samples, the detector might recognize false positives, leading to an effect similar to the drift problem.

Kalal et al. [8] proposed a semisupervised learning combining both approaches, each representing a different component of their Tracking-Learning-Detection (TLD) framework. It uses a motion model tracker to fill the detector training set with new appearances. A detector corrects tracker failures replacing the template. The complementary nature of the components and the results achieved suggest that this is a promising combination. Not all tracker responses are used for retraining. The detector similarity function is used to validate them and as well to select the system output among tracker and detector responses. In contrast, Rosenberg et al. [7] show that their detector-independent metric outperforms the detector similarity function in a re-training step because the learning and detection failure cases tend to be distinct. Similarly, our proposal is to use detector independent methods for validation and selection in TLD framework.

ALIEN is a semisupervised tracker by detection proposed by Pernici and Bimbo [6]. They perform tracking through weak alignment of SIFT [10] points and detect occlusions by monitoring keypoints of the context, i.e. the region around the object. If the number of context keypoints that invade the object region is greater than a threshold, an occlusion has occurred. In these cases, the detections are not used to update the set of SIFT points. Since the drift problem is strongly related to background insertion in the object region, this is a

Authors thank to FAPEMIG, CAPES and UFJF for funding.

suitable technique for tracker validation in TLD framework.

Schwartz and Davis [11] propose the use of a rich descriptor for appearance-based tasks, with focus on person recognition. This descriptor is a combination of low level features: color, texture and shape. Considering different aspects, the system is capable to deal with different challenges commonly found in appearance-based tasks. Using color and texture, for example, the system can overcome occlusions and deformations, situations in which shape is not useful. On the other hand, color is not a useful aspect when illumination variation occurs. In our work, we propose the use of this rich descriptor to obtain a reliability score for response selection in TLD framework, exploring its generality power.

The main contribution of this work is a mediator that validates a greater variety of object appearances for detector retraining and selects good motion tracker responses avoiding improper re-initializations, as the validation and selection decisions are not influenced by the detector itself. As a result, our tracker outperforms the original, in particular by estimating the correct position more often.

2. PROPOSED METHOD

An overview of our method is depicted in Fig. 1. System samples and responses are sub-images delimited by bounding boxes (BB) possibly containing the object being tracked. Thus, all components only exchange BB coordinates containing the sub-image in the respective frame. The object sample is a BB in the first frame given by user to initialize the tracker and train the detector. A *template model*, used exclusively by the tracker, contains a reference frame (the previous one) and its BB. An *object model* comprises a list of sample BBs collected so far (training set) and two similarity functions: one for the detector (uses whole training set) and the other for the selector/validator (conservatively uses 50% of the training set). We define the selector (Fig. 1(a)) as the component responsible for giving the estimated object's BB for each frame f_t. For that, it receives one BB estimated from the motion tracker and a list of BBs estimated as likely containing the object from the detector. Then, it selects one of the inputs, if any, to be the system response. The selected response is also used to update the motion tracker template (Fig. 1(b)). The validator is responsible for deciding if the tracker estimation can be used for re-training the detector. If current tracker estimation is considered invalid, the object model remains the same. Otherwise, the learning component uses it as reference to verify the detector's list of sample BBs. The detector responses considered incorrect are sent to the model to improve future detections.

In the original method, the selection component uses the object model to choose the final answer. We argue that it tends to benefit detector responses because they are based on the same similarity functions and samples. It also affects future tracker estimations since the final response replaces the old

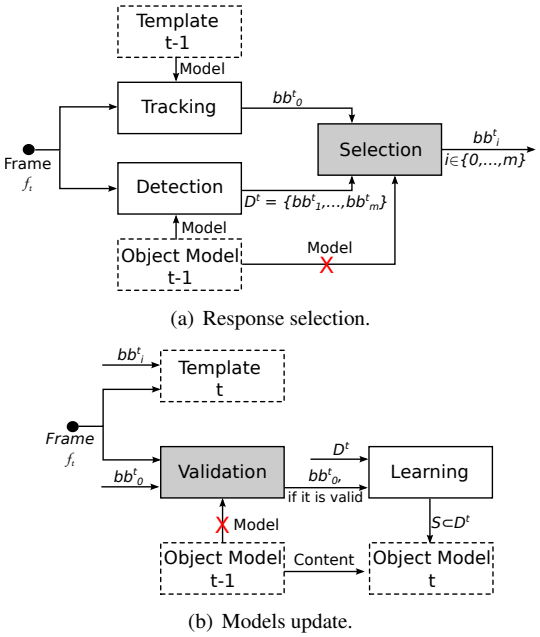

Fig. 1. Overview of our method where the validation and selection make decisions only with their own knowledge. The red crosses indicate connections of the original TLD removed in our proposal.

template. In the validator, the reliability score of the tracker response is given by the object model too. This way, the validator uses the detector knowledge to decide when the detector could be re-trained. In our point of view, the selection and validation components have to be defined independently of the detector and tracker in order to get the best of each approach. In other words, both should have their own metrics and samples to make decisions. So, we keep the same tracker, detector and learning method from the original, but propose new validation and selection methods as shown in the following subsections.

2.1. Response selection

The selector receives one BB from the tracker and a list of BBs from the detector. Tracker and detector might not always provide BBs, meaning the object is invisible. If no BBs are given, the selector has no output too, i.e. object not found. Otherwise it selects one of the inputs as the system response.

Using the first sample, the selector has to decide which response presents a sound appearance of the tracked object at each frame. But the object can assume different appearances throughout the video and a good selector should take them into account to accept the most likely and reject the least. To address the problem, we use a rich descriptor combining color, texture and shape, similar to Schwartz et al. [11].

Color information is represented by an intensity histogram $\vec{h_c} \in \mathbb{R}^{256}$ using one channel. Texture is represented by four local binary pattern histograms (LBP [12]),

$\vec{h_{l1}}, \vec{h_{l2}}, \vec{h_{l3}}, \vec{h_{l4}} \in \mathbb{R}^{256}$, resulting in 1024 features. Shape is described using a one-dimensional histogram of oriented gradient (HOG [13]) $\vec{h_s} \in \mathbb{R}^{16}$. Each histogram is normalized using L_2 norm. The descriptor is a set which contains these features. We store the descriptor D' of the initial sample and compute new descriptors D for each response sent to the selector at runtime.

The goal of the selector is to pick the largest similarity response to the first descriptor. Since each feature has a different number of bins, we compute three individual similarity values whose votes are evenly weighted. The similarity between two histograms is their dot product. For texture, the similarity is the average of the LBP histograms:

$$S_l(D) = \frac{\vec{h_{l1}} \cdot \vec{h'_{l1}} + \vec{h_{l2}} \cdot \vec{h'_{l2}} + \vec{h_{l3}} \cdot \vec{h'_{l3}} + \vec{h_{l4}} \cdot \vec{h'_{l4}}}{4}.$$

The final similarity $S(D)$ is the average between the votes

$$S(D) = \frac{\vec{h_c} \cdot \vec{h'_c} + S_l(D) + \vec{h_s} \cdot \vec{h'_s}}{3},$$

from which the best choice keeps a little of every aspect or is very similar in some of them.

2.2. Tracker validation

Trackers by motion model generally fail when dealing with insertion of background in the template. TLD's tracker includes a *failure detector* to identify abrupt changes such as fast occlusions, but gradual changes still represent a challenge considering the tracker's lack of memory. Since it performs local searches, its responses are somewhat close to the last one given. For this reason, template degradation often come from elements in the object neighborhood. To prevent corrupted samples from being added to the detector training set, we keep a record of context features as proposed by Pernici and Bimbo [6]. The context is a region surrounding the object's BB within a fixed margin.

In the first frame f_0, we extract SIFT points from the context, defined by a margin m, forming the initial set C_0. At each new frame f_t, we match feature points extracted from the tracker's result bb_t with the previous context features C_{t-1} (Alg. 1). The matching follows the same steps as proposed in [10]. 1NN(d) and 2NN(d) are, respectively, the first and second nearest neighbors of the descriptor d in C_{t-1}. If the number of points matched with the context exceeds a threshold n_o, the tracked sub-image is not reliable. Otherwise, bb_t is valid and used for detector re-training (Fig. 1(b)). Context features S'_t are collected around bb_t, within the margin m, for future matchings. They are kept for $l > 1$ frames. The occlusion features D_t formed by accumulated matched points, on the other hand, are kept indefinitely to make sure they will not be detected as object after l frames. When $|D_t|$ reaches a maximum size, some features are randomly removed to control set growth. Thus, the new context set C_t is the union of D_t and the l most recent context features $\{S'_\tau \mid \tau > t - l\}$.

Algorithm 1: Tracker validation

Data: Tracker response $bb_t = (x_1, y_1, x_2, y_2)$.
Result: Response validity v (true or false).

begin
- $v \leftarrow false$;
- //Extract SIFT points from object
- $P \leftarrow \{(x, y) \in \mathbb{Z}^2 \mid x_1 \le x \le x_2, y_1 \le y \le y_2\}$;
- $S_t \leftarrow \{(p, d) \in P \mid p$ is a keypoint, d is the descriptor$\}$;
- //Compute matching features with C_{t-1}
- $C_t^* \leftarrow \{(p, d) \in S_t \mid \frac{\|d - 1\text{NN}(d)\|}{\|d - 2\text{NN}(d)\|} < \lambda_c\}$;
- **if** $|\ C_t^* \le n_o$ **then**
  - $v \leftarrow true$; //Reliable response
  - //Extract SIFT points from current context
  - $P' \leftarrow \{(x, y) \in \mathbb{Z}^2 \setminus P \mid x_1 - m \le x \le x_2 + m,$
    $y_1 - m \le y \le y_2 + m\}$;
  - $S'_t \leftarrow \{(p, d) \in P' \mid p$ is a keypoint, d is the descriptor$\}$;
  - //Store occlusion features
  - $D_t \leftarrow D_{t-1} \cup C_t^*$;
  - **if** $|\ D_t\ | > n_d$ **then** $RandomRemoval(D_t, n_d)$;
  - $C_t = D_t \cup (\bigcup_{\tau=t-l}^{t} S'_\tau)$; //Context set update

3. EXPERIMENTAL RESULTS

3.1. Evaluation protocol

For evaluation purposes, we use the TLD dataset [14] which contains 10 sequences with different objects. They also provide the expected answers, or ground truth (GT), for each sequence. A response is considered correct if both the response and corresponding GT are visible, and the ratio between their intersection and union is greater than 25%. For each sequence, we compute exactly the same metrics used in the original work. Precision P is the number of correct responses divided by the number of visible responses. Recall R is the number of correct responses divided by the number of visible GTs. F-measure is given by $F = \frac{2PR}{(P+R)}$ and it is the main metric for comparing results, since it combines precision and recall. Overall performance is given by the mean performance weighted by the sequence's number of frames.

3.2. Results

We compare the original and the proposed selector/validator. We fully implemented the TLD to conduct the experiments. The components were combined into four variations: original selector and validator (*os-ov*), original selector and proposed validator (*os-pv*), proposed selector and original validator (*ps-ov*), proposed selector and validator (*ps-pv*). For every variation, we tested different tracker and detector settings such as different window sizes for the tracker optical flow. In the following results, we used the setting with the best overall performance for each variation. For the validator, we use $n_o = 2$, $m = 20$, $n_d = 1500$ and $l = 10$ as proposed in the ALIEN [6] and $\lambda_c = 0.7$ that empically gave the best results.

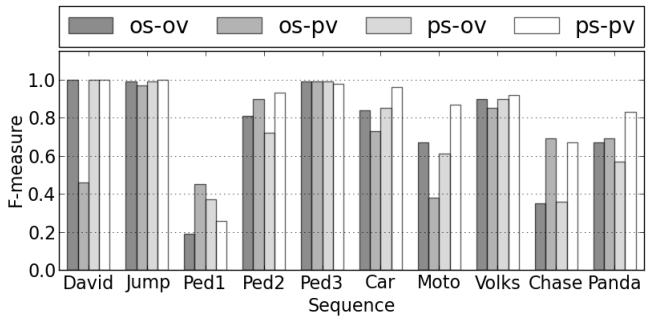

Fig. 2. F-measure by sequence and variation. *os* is the original selector and *ps* the proposed one. *ov* is the original validator and *pv* the proposed one.

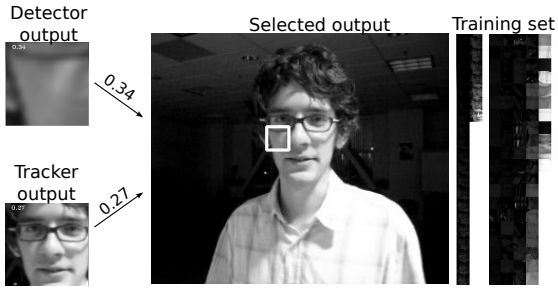

Fig. 3. Example of incorrect tracker re-initialization in David sequence using *os-pv*. Even with a good training set, the similarity value for the correct tracker output is smaller. It is important to remember that *os* uses only the first 50% of the samples.

In the first experiment, we analyze the effect of replacing each component in the original method using the aforementioned variations (Fig. 2). For some sequences (*David, Jumping, Car, Motocross, Volkswagen*) the f-measure decreases considerably when replacing only the validator (*os-pv*). A possible cause of this low performance is incorrect re-initialization in the selector. The Fig. 3 shows an example during the David sequence, where the selector picks the detector's response even when the tracker gives a better one. This leads to a template change in the tracker, affecting its future estimations and, consequently, the validator's tasks. This occurs in other sequences and illustrates how the error from a component may cascade through the system. As such, if the validator cannot accept good samples for detector retraining, the detector also cannot send good responses to the selector. We argue that all modules must take their decisions independently to reduce this coupling problem. Notice that even in the sequences where the performance decreased by replacing one of the components, using both of our decoupled proposals *ps-pv* give better results.

Detailed results of *os-ov* and *ps-pv* are given in Tab. 1. Note that our proposal increased the recall of almost all the sequences without compromising the precision. This means that our method is capable of tracking farther. In particular, sequences *Motocross*, *Carchase* and *Panda* had a meaningful improvement. The objects in *Motocross* and *Carchase* change their pose throughout the sequence. In *Panda*, the object deforms constantly. This variety of appearances becomes a failure case of *os-ov* since its validator, by specifically using normalized cross-correlation (NCC), does not include new appearances that are very far from the known ones. By choosing NCC, *os-ov* imbues to its validator a property from the detector that is not found in the tracker. This goes against the main goal of using the tracker's property of accepting new appearances to improve the detector. In *ps-pv*, by assigning to the validator a less partial method, it is possible to obtain a richer training set that accepts more distinct appearances. We atribute this validator property, along with proper selector activity, to the increase in performance. We also compare our method with the state-of-the-art ALIEN results [6]. Although our results are close to theirs, they are still lower, even considering that ALIEN employs a stricter criterion for correct BBs (intersection-union ratio greater than 50%). However, our method gives better precision which might be an interesting characteristic for applications that need reliable BB's.

Video	Frames	*os-ov* (TLD) P / R / F	*ps-pv* (our) P / R / F	ALIEN P / R / F
David	761	**1.00 / 1.00 / 1.00**	**1.00 / 1.00 / 1.00**	0.99 / 0.98 / 0.99
Jump	313	1.00 / 0.98 / 0.99	**1.00 / 1.00 / 1.00**	0.99 / 0.87 / 0.92
Ped1	140	0.39 / 0.13 / 0.19	0.37 / 0.21 / 0.26	**1.00 / 1.00 / 1.00**
Ped2	338	0.99 / 0.68 / 0.81	**0.90 / 0.97 / 0.93**	**0.93 / 0.92 / 0.93**
Ped3	184	**0.99 / 1.00 / 0.99**	0.96 / 1.00 / 0.98	1.00 / 0.90 / 0.95
Car	945	0.87 / 0.81 / 0.84	0.93 / 0.99 / 0.96	**0.95 / 1.00 / 0.98**
Moto	2665	0.78 / 0.58 / 0.67	**0.88 / 0.86 / 0.87**	0.69 / 0.81 / 0.74
Volks	8576	0.88 / 0.92 / 0.90	0.93 / 0.91 / 0.92	**0.98 / 0.89 / 0.93**
Chase	9928	0.96 / 0.21 / 0.35	0.95 / 0.51 / 0.67	**0.73 / 0.68 / 0.70**
Panda	3000	0.64 / 0.70 / 0.67	**0.80 / 0.87 / 0.83**	-
Mean	26850	0.88 / 0.59 / 0.64	0.92 / 0.76 / 0.81	**0.84 / 0.80 / 0.82**

Table 1. Comparison between *os-ov* (implemented TLD) and *ps-pv* (our proposal). Cells on bold represents the highest f-measure.

4. CONCLUSION

We presented a novel method for object tracking based on independent selector/validator in TLD, i.e. these components use only their own knowledge to make decisions. Our results showed that replacing one non-independent component at a time is not effective. But replacing both outperforms the original method. This shows that the components are strongly coupled in the framework. Besides, we show that our method tracks farther, specially for objects with high appearance variations throughout the sequence. See [15] for further information about our method for multiple tracker mediation.

Our method has some drawbacks, even giving better results. The validator is vulnerable to the tracker re-initializations, since it uses the context of the tracker responses. Future works may include improvements in the validator in order to recover to older and more reliable contexts. The selector might be improved by learning a distance metric from the few labeled samples for each aspect or using more aspects.

5. REFERENCES

[1] Bruce D. Lucas and Takeo Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, San Francisco, CA, USA, 1981, vol. 2 of *IJCAI'81*, pp. 674–679, Morgan Kaufmann Publishers Inc.

[2] Jianbo Shi and Carlo Tomasi, "Good features to track," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 1994, pp. 593–600.

[3] Iain Matthews, Takahiro Ishikawa, and Simon Baker, "The template update problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 810–815, June 2004.

[4] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas, "Forward-backward error: Automatic detection of tracking failures," in *20th International Conference on Pattern Recognition (ICPR)*, Aug 2010, pp. 2756–2759.

[5] Mustafa Ozuysal, Pascal Fua, and Vincent Lepetit, "Fast keypoint recognition in ten lines of code," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007, pp. 1–8.

[6] Federico Pernici and Alberto Del Bimbo, "Object tracking by oversampling local features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2538–2551, Dec 2014.

[7] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman, "Semi-supervised self-training of object detection models," in *Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTIONS)*, Jan 2005, vol. 1, pp. 29–36.

[8] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, July 2012.

[9] Dae-Youn Lee, Jae-Young Sim, and Chang-Su Kim, "Multihypothesis trajectory analysis for robust visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 5088–5096.

[10] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[11] William Robson Schwartz and Larry S Davis, "Learning discriminative appearance-based models using partial least squares," in *XXII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, Oct 2009, pp. 322–329.

[12] Timo Ojala, Matti Pietikäinen, and David Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51 – 59, 1996.

[13] Navneet Dalal, Bill Triggs, and Cordelia Schmid, "Human detection using oriented histograms of flow and appearance," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision*, pp. 428–441. Springer, Berlin, Heidelberg, May 2006.

[14] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010*, June 2010, pp. 49–56.

[15] Helena de Almeida Maia, "A Mediator for Multiple Trackers in Long-term Scenario," M.S. thesis, Universidade Federal de Juiz de Fora, Juiz de Fora-Brazil, 2016.