FAST AND ACCURATE VIDEO ANNOTATION USING DENSE MOTION HYPOTHESES

Loïc Fagot-Bouquet, Jaonary Rabarisoa, Quoc Cuong Pham

CEA, LIST, LVIC, Point Courrier 173, F-91191 Gif-sur-Yvette, France
loic.fagot-bouquet@cea.fr, jaonary.rabarisoa@cea.fr, quoc-cuong.pham@cea.fr

ABSTRACT

Building large video datasets is a crucial task for many applications but is also very expensive in practice. In order to avoid annotating all the frames, the annotations from the labeled frames can be propagated using an offline tracker for each object. Following methods based on dynamic programming and eventually distance transforms, we introduce a new penalization which favors some given displacements between two frames without increasing the complexity of the optimization. In order to speed up this step we also propose to use an exact coarse to fine process. Experimental results show that the proposed energy performs better than previous ones and that our exact coarse to fine optimization leads to a significant speed-up in some scenarios.

Index Terms— video annotation, offline tracking, coarse to fine, distance transform, dynamic programing

1. INTRODUCTION

Large video datasets are often needed in order to evaluate or train algorithms in computer vision. However, producing annotated video datasets is really expensive as they can include many thousand (even millions) frames. Some interactive annotation tools have been designed in recent years to facilitate this task, both for image and video annotations [1, 2, 3, 4, 5, 6]. Concerning videos and bounding boxes annotations for each object, the main idea is to avoid annotating all the frames by propagating the boxes from the labeled ones, using linear interpolation for example. More generally this can be done by tracking offline the objects one wants to annotate. It has been shown in [6] that annotating multiple objects in a video is very difficult for users whereas dealing with one object at a time yields better results (and is also preferred by the annotators). In this context, the user keeps giving new annotations of the object until its trajectory is correctly retrieved. Therefore we focus in this paper on the scenario where one must interactively track a single object without any assumption on it but some of its locations over the video.

In order to use all the available information, offline tracking is often reduced to an optimization problem. The estimated path p^* over T frames is therefore the one which minimizes an energy function E generally written as:

$$E(p) = U_1(p_1) + \sum_{t=2}^{T} [U_t(p_t) + \lambda d(p_t, p_{t-1})] \qquad (1)$$

where U_t are the scores given by an appearance model which favors the locations with an appearance similar to the tracked object, d is a function favoring smooth trajectories and λ determines a trade-off between these two terms. A known location of the object in the frame t can be taken into account by attributing an infinite score for the other locations in U_t. If K possible locations per frame are allowed, then the optimization can be done in $O(K^2T)$ using dynamic programming. Due to this quadratic complexity, paths going through a small number of best locations (according to the appearance model) per frame were only considered in [7]. In [8] the energy was optimized over all paths in $O(T)$ by making a strong assumption on the appearance model which is not easily satisfied.

A great improvement in this optimization process has been done in [5] and [9]. These two papers explained that the dynamic programming approach could be done over all the paths in $O(NT)$, with N the resolution of the video. In fact, general distance transforms (introduced in [10]) are used in these papers in order to significantly reduce the complexity. Some assumptions on the function d are required but the usual functions (l_1 norm, used in [9], and l_2 norm squared, used in [5]) can still be applied (more details can be found in [10]). However, the d function was introduced to favor smooth paths but the l_1 norm and l_2 norm squared also penalize the length of the paths. The l_2 norm squared greatly favors linear paths between the given locations of the object whereas no such assumption is done on its motion.

Our contribution in this paper is to propose a new energy formulation which generalizes the previous one without increasing the complexity of the optimization process. This new energy function is designed to use the estimated motion of the object over the frames and favors paths which are relevant with these informations. Secondly, we propose to solve more efficiently the introduced energy using an exact coarse to fine approach which leads to a significant speed-up in some scenarios. Using a coarse to fine strategy to solve dynamic programming problems have been proposed in [11] but without considering distance transforms.

2. PROPOSED ENERGY

Penalizing displacements between consecutive frames in (1) results in discrediting elongated paths and favoring linear paths between given annotations (in the case of the l_2 norm squared). We propose instead to penalize the inconsistency between the considered path and some motion hypotheses by introducing the following energy:

$$E_f(p) = U_1(p_1) + \sum_{t=2}^{T} [U_t(p_t) + \min_{u \in f_t(p_t)} \lambda d(u, p_{t-1})]$$

where d a function that allows us to use distance transforms and for each frame t and location p_t, $f_t(p_t)$ is a set of locations in frame $t-1$ ($f_t(p_t)$ can be seen as a set of hypotheses on the backward motion of an object located in p_t). In the rest of this paper we assume for the complexity results that $|f_t(p_t)| = O(1)$ with respect to the frame resolution N.

This energy can be minimized using a dynamic programming approach and distance transforms as described in [5, 9]. Let us denote by $C_t(p_t)$ the minimal energy value of a path beginning at the first frame and ending in p_t on the frame t. All the terms C_t can be computed with a recursive formula:

$$C_{t+1}(p_{t+1}) = U_{t+1}(p_{t+1}) + M_{C_t}$$

$$M_{C_t}(p_{t+1}) = \min_{p_t} [C_t(p_t) + \min_{u \in f_{t+1}(p_{t+1})} \lambda d(u, p_t)] \quad (2)$$

An optimal path can be retrieved in $O(T)$ from the terms C_t and the indexes $\pi_{t+1}(p_{t+1})$ which achieve the minimum over p_t in (2):

$$\begin{cases} p_T^* = \underset{p_T}{argmin}\, C_T(p_T) \\ p_t^* = \pi_{t+1}(p_{t+1}^*) \end{cases}$$

Considering the distance transform DT_{C_t} and the related indexes α_{t+1} (computed both in $O(N)$ according to [10]) given by:

$$DT_{C_t}(u) = \min_{p_t} C_t(p_t) + \lambda d(u, p_t)$$

$$\alpha_{t+1}(u) = \underset{p_t}{argmin}\, C_t(p_t) + \lambda d(u, p_t)$$

we have then:

$$\begin{aligned} M_{C_t}(p_{t+1}) &= \min_{p_t} [C_t(p_t) + \min_{u \in f_{t+1}(p_{t+1})} \lambda d(u, p_t)] \\ &= \min_{u \in f_{t+1}(p_{t+1})} \min_{p_t} C_t(p_t) + \lambda d(u, p_t) \\ &= \min_{u \in f_{t+1}(p_{t+1})} DT_{C_t}(u) \end{aligned}$$

$$\pi_{t+1}(p_{t+1}) = \alpha_{t+1}(\underset{u \in f_{t+1}(p_{t+1})}{argmin}\, DT_{C_t}(u))$$

Since we assume that $|f_t(p_t)| = O(1)$, C_{t+1} and π_{t+1} can be computed from C_t in $O(N)$. Therefore, we are able to optimize E_f in $O(TN)$.

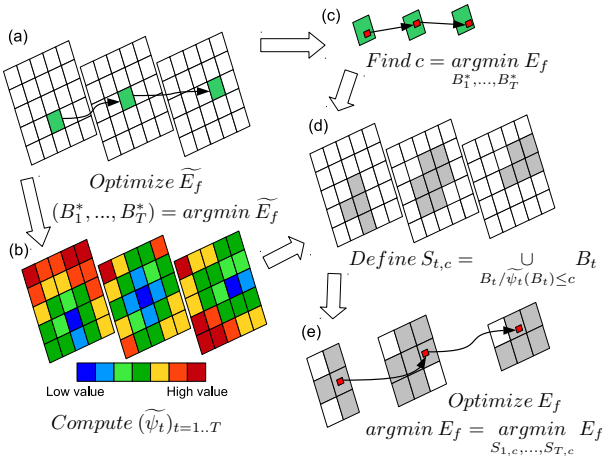


Fig. 1. Exact coarse to fine approach. The coarse energy is optimized (a) and min-marginals are computed (b). The value c is determined by optimizing E_f over the best coarse path (c) and used to reduce the set of possible paths (d). E_f is minimized over these paths, yielding a global solution (e).

3. EXACT COARSE TO FINE OPTIMIZATION

We propose in this section to optimize more efficiently E_f using an exact coarse to fine approach.

The main idea is to quickly solve our problem at a coarse level and use the retrieved information to limit the regions where the object trajectory can go through. First, we need to define a coarse energy $\widetilde{E_f}$ (written the same way as E_f) by considering paths of $s \times s$ square blocks of pixels instead of pixelwise paths. $\widetilde{E_f}$ should be solved rapidly using distance transforms (as explained in previous section) and should satisfy:

$$\widetilde{E_f}(B_1, ..., B_T) \leq \min_{p_1 \in B_1, ..., p_T \in B_T} E_f(p_1, ..., p_T) \quad (3)$$

which means that the value of a path of blocks is a lower bound of the value of any path going through these blocks.

$\widetilde{E_f}$ is optimized, yielding an optimal path $B_1^*, ..., B_T^*$. We denote by $c = E_f(p_{B_1}^*, ..., p_{B_T}^*)$ the minimum value of a pixelwise path going through $B_1^*, ..., B_T^*$ (found by optimizing E_f over $B_1^*, ..., B_T^*$). Usual coarse to fine approaches would use c as an approximation of the minimal cost and stop here, but $\widetilde{E_f}$ can be used differently to retrieve the exact solution (thanks to the assumption (3)).

We need to define the notion of min-marginals, denoted by $\psi_t(p_t)$, as the minimum value of a path going through the location p_t in frame t. As done in [5], it is possible to compute the ψ_t for E_f in $O(TN)$ (see section 4 for details). Therefore the min-marginals related to $\widetilde{E_f}$, $\widetilde{\psi_t}$, are computed and we consider for each frame t the set:

$$S_{t,c} = \underset{B/\widetilde{\psi_t}(B) \leq c}{\cup} B$$

The energy E_f is then minimized over the sets $S_{1,c}, ..., S_{T,c}$ with the guarantee of finding the minimal cost $E_f(p_1^*, ..., p_T^*)$. The minimal cost is found because if we consider a block B_t with $B_t \notin S_{t,c}$ and a path $p_1, ..., p_t, ..., p_T$ with $p_t \in B_t$, then we have:

$$E_f(p_1, ..., p_T) \geq \psi_t(p_t) \geq \widetilde{\psi_t}(B_t) > c = E_f(p^*_{B_1}, ..., p^*_{B_T})$$

The algorithm from [10] can be easily adapted to compute distance transform over regions of different sizes and locations (in order to optimize E_f over $B_1^*, ..., B_T^*$ and $S_{1,c}, ..., S_{T,c}$). However, can we find an energy $\widetilde{E_f}$ that fulfills the needed requirements ? This is possible when d in E_f is chosen as the l_2 norm squared and some details are given in the next section.

4. TECHNICAL DETAILS

4.1. Min-marginals

The min-marginals related to E_f can be computed in $O(TN)$ as done in [5] for E. Using the notations from [5], one can observe that $\psi_t(p_t) = \overrightarrow{C_t}(p_t) + \overleftarrow{C_t}(p_t) - U_t(p_t)$ where the $\overrightarrow{C_t}$ terms are equal to the C_t ones introduced in section 2 and $\overleftarrow{C_t}(p_t)$ stands for the minimum value of a path between the frame t and the last frame which begins in p_t. These terms can be computed, as we did for C_t, using a recursive formula and distance transforms:

$$\overleftarrow{C_{t-1}}(p_{t-1}) = U_{t-1}(p_{t-1}) + DT_G(p_{t-1})$$

with $DT_G(p_{t-1}) = \min_u G(u) + \lambda d(u, p_{t-1})$

and $G(u) = \min_{p_t / u \in f_t(p_t)} \overleftarrow{C_t}(p_t)$

Therefore, the terms $\overleftarrow{C_t}$ can be computed in $O(TN)$, and the same result holds for the min-marginals ψ_t.

4.2. Coarse energy

In order to have an energy $\widetilde{E_f}$ that satisfies (3), $\widetilde{U_t}$, $\widetilde{d}$ and $\widetilde{f_t}$ can be chosen as following. We first consider coarse scores $\widetilde{U_t}$ which satisfy:

$$\widetilde{U}_t(B_t) \leq \min_{B_t} U_t \tag{4}$$

and we use $\widetilde{U}_t(B_t) = \min_{B_t} U_t$ in practice. $\widetilde{f_t}$ is defined as:

$$\widetilde{f}_t(B_t) = \{B \in \mathcal{B} \ : \ B \cap f_t(B_t) \neq \emptyset\} \tag{5}$$

where $\mathcal{B}$ stands for the set of blocks over a frame. $\widetilde{d}$ is chosen such that:

$$\widetilde{d}(B_t, B_{t-1}) \leq \min_{p_t \in B_t, p_{t-1} \in B_{t-1}} \lambda d(p_t, p_{t-1}) \tag{6}$$

Such a function $\widetilde{d}$ can be found in the case where d is set to the l_2 norm squared. We consider:

$$\widetilde{d}(B, B') = \widetilde{d_{1d}}(B_x - B'_x) + \widetilde{d_{1d}}(B_y - B'_y)$$

$$\widetilde{d_{1d}}(u) = \lambda \times min((su)^2, \ (s(u-1)+1)^2, \ (s(-u-1)+1)^2)$$

Considering (4), (5) and (6), one can check that $\widetilde{E_f}$ satisfies (3). Moreover, it is still possible to compute distance transforms with $\widetilde{d}$ by computing the minimum of three distance transforms.

5. EXPERIMENTATION

5.1. Implementation

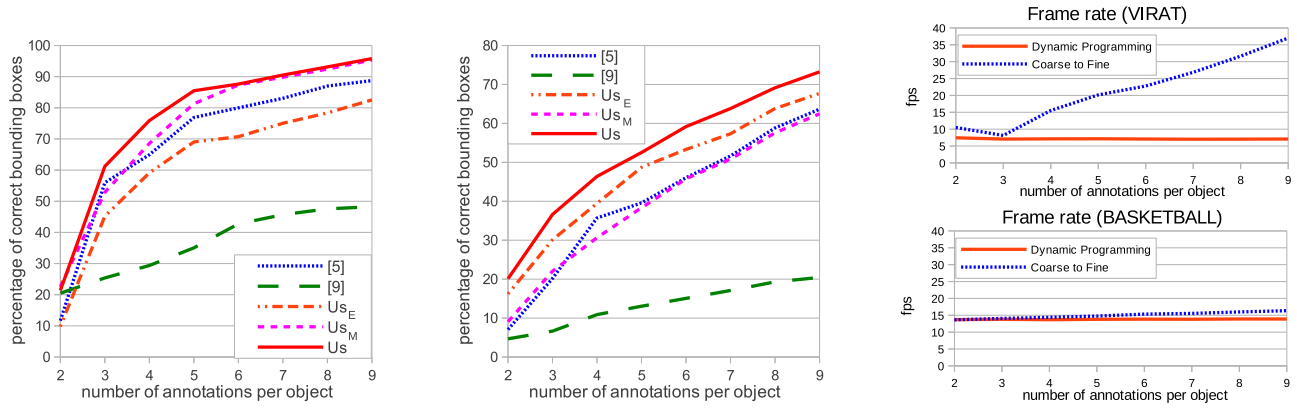
Our approach was implemented in C++ and tested on a server (16 cores at 3.1 GHz and 64 GB of RAM but using less than 2 GB). We used a simple bag of features based appearance model (with a linear SVM as described, for example, in [12]) and optical flows to compute the f_t functions.

In details, dense SIFT features are computed and a large number of bags of features are determined from a small set of frames. The main advantages of this model are that the scores can be quickly evaluated accross the video and many calculations can be precomputed independently of the tracked object. Concerning the f_t functions, we only consider the case where $f_t(p_t)$ is restrained to the position p_{t-1} estimated by backward optical flows (with respect to the estimated object size).

5.2. Experimental setting

We have compared our algorithm with the two most related works ([9] and the offline tracker from [5]) on the same datasets than [5]. About twenty trajectories from the VIRAT dataset [13] were used, and about forty from the more difficult basketball match. The resolution of the videos from the VIRAT dataset was reduced to 1080×720 and the framerate set to 5 fps. We kept the original resolution (720×480) of the basketball video and its framerate (30 fps). Considering these trajectories, this leads to more than 35000 bounding boxes. The coarse to fine optimization was also tested, using blocks of size 10×10 on VIRAT and 5×5 on the basketball one. In all cases the first and last frames are given and we incrementally add new annotations by selecting the more distant frame from the given ones.

Fig. 3. Datasets used. Left: VIRAT, right: BASKETBALL.



(a) Success rate (VIRAT) (b) Success rate (BASKETBALL) (c) Optimization speed

Fig. 2. (a) and (b): Success rate (percentage of correctly retrieved bounding boxes). Our appearance model was tested with the energy E_f (denoted by Us) and the usual energy E (Us$_E$). E_f was also considered without any appearance model, using only estimated motion (Us$_M$). (c): Frame rates for the dynamic programming optimization and the coarse to fine approach.

Quantitative analysis was done using success rate (percentage of correctly retrieved bounding boxes) and average center location error. A ground truth bounding box b_t^{gt} is considered correctly retrieved by a bounding box b_t when $\frac{b_t \cap b_t^{gt}}{b_t \cup b_t^{gt}} > 0.5$, and its center location error is defined as the Euclidian distance in pixels between the centers of b_t^{gt} and b_t.

Selecting a good λ parameter (within the energies E and E_f) is crucial in order to establish a fair comparison among these methods. For each dataset and each method, several parameters were tried and the best one was chosen (according to success rate values). The values $\lambda = 10^i$ $(-2 \leq i \leq 2)$ were tried for our methods and the one from [5], whereas the values $\lambda = 50 \times 10^i$ $(-2 \leq i \leq 2)$ were used for the algorithm from [9] (since the default λ parameter was fixed to 50 in [9]).

5.3. Results

The scores of the different approaches are shown in figure 2 and table 1. We achieve results close to [5] using our appearance model and the usual energy E (Us$_E$). Our energy E_f without any appearance model (Us$_M$) correctly retrieves about the same number of bounding boxes as [5] or the usual energy E with our appearance model (Us$_E$), and yields better results regarding center locations. However, combining these two sources of information in E_f (denoted by Us) outperforms in most cases all the tested methods. Therefore, adding an estimation of the motion in the usual energy improves significantly the results. The lower results of the method proposed in [9] are mainly due to the l_1 norm penalization (which does not favor smooth paths contrary to the l_2 norm squared).

	VIRAT					BASKETBALL				
	[5]	[9]	Us$_E$	Us$_M$	Us	[5]	[9]	Us$_E$	Us$_M$	Us
2	309	**134**	283	240	258	168	139	140	**95.5**	104
3	44.5	44.0	55.7	34.1	**30.0**	99.4	126	81.3	**61.1**	65.4
4	26.3	40.4	35.6	16.5	**12.9**	68.1	101	57.8	49.8	**43.2**
5	17.0	34.2	18.5	8.72	**7.23**	51.9	90.3	42.6	40.3	**38.1**
6	15.5	24.0	17.0	7.09	**6.73**	40.4	83.5	38.4	32.6	**29.7**
7	14.0	21.0	14.5	6.46	**6.20**	36.5	73.5	36.0	28.2	**27.4**
8	10.3	20.2	11.5	6.02	**5.75**	28.1	62.2	25.8	21.2	**18.3**
9	9.67	17.2	9.38	5.38	**5.08**	21.6	55.8	18.8	16.5	**13.7**

Table 1. Average center location error (best in bold, second best underlined) given the number of annotations per object.

As shown in figure 2, the exact coarse to fine optimization is faster than the standard one on the VIRAT dataset (with a speed-up ratio up to 5) but is not significantly faster on the basketball match. This can be easily explained as this video includes a lot of similar players which leads to a lot of promising paths. For this reason the coarse to fine method does not succeed in reducing the set of possible paths in this situation.

6. CONCLUSION

In this paper we have proposed to annotate videos using a new energy formulation for the offline tracker which favors trajectories relevant to some motion hypotheses accross the video. Our approach outperforms previous energy functions penalizing displacements and is still optimized in linear time. The proposed coarse to fine optimization achieves significant speed-up in some scenarios without deterioring the computation time of the difficult ones.

As we have only used multiple displacements (when the set $f_t(p_t)$ is not reduced to a single location) for the coarse energy $\widetilde{E_f}$, future work should consider this possibility directly for the energy E_f. An active learning process similar to [5] could also be designed for the proposed energy formulation.

7. REFERENCES

[1] A. Sorokin and D. Forsyth, "Utility data annotation with Amazon Mechanical Turk," *Computer Vision and Pattern Recognition Workshops*, pp. 1–8, 2008.

[2] J. Yuen, B. Russell, and A. Torralba, "LabelMe video: Building a video database with human annotations," *International Conference on Computer Vision*, pp. 1451–1458, 2009.

[3] A. Yao, J. Gall, C. Leistner, and L. Van Gool, "Interactive object detection," *Conference on Computer Vision and Pattern Recognition*, pp. 3242–3249, 2012.

[4] L. Von Ahn, R. Liu, and M. Blum, "Peekaboom: a game for locating objects in images," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2006.

[5] C. Vondrick and D. Ramanan, "Video annotation and tracking with active learning," *Advances in Neural Information Processing Systems*, pp. 1–9, 2011.

[6] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer*, 2013.

[7] Y. Wei, J. Sun, X. Tang, and H. Shum, "Interactive Offline Tracking for Color Objects," *International Conference on Computer Vision*, pp. 1–8, 2007.

[8] S. Uchida, I. Fujimura, H. Kawano, and Y. Feng, "Analytical dynamic programming tracker," *Asian Conference on Computer Vision*, 2010.

[9] S. Gu, Y. Zheng, and C. Tomasi, "Linear time offline tracking and lower envelope algorithms," *International Conference on Computer Vision*, , no. 2, 2011.

[10] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," *Theory of computing*, vol. 8, pp. 415–428, 2012.

[11] C. Raphael, "Coarse-to-fine dynamic programming," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1379–1390, 2001.

[12] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in *International Conference on Computer Vision*, 2009.

[13] S. Oh, A. Hoogs, and A. Perera, "A large-scale benchmark dataset for event recognition in surveillance video," *Computer Vision and Pattern Recognition*, , no. 2, 2011.