Motion-sketch based Video Retrieval using a Trellis Levenshtein Distance

Rui Hu and John Collomosse
Centre for Vision, Speech and Signal Processing (CVSSP)
University of Surrey, Guildford, Surrey, U.K.
{*R.Hu* | *J.Collomosse*}*@surrey.ac.uk*

Abstract

We present a fast technique for retrieving video clips using free-hand sketched queries. Visual keypoints within each video are detected and tracked to form short trajectories, which are clustered to form a set of space-time tokens summarising video content. A Viterbi process matches a space-time graph of tokens to a description of colour and motion extracted from the query sketch. Inaccuracies in the sketched query are ameliorated by computing path cost using a Levenshtein (edit) distance. We evaluate over datasets of sports footage.

1. Introduction

Techniques to search video databases are increasingly important in a media rich world. Video Retrieval is predominantly addressed by searching metadata tags, yet keyword based retrieval can be semantically ambiguous, and often lacks rich descriptive power for actions. Querying by Visual Example (QVE) can mitigate these problems, however most video QVE techniques require a photo-real query (an image [18], or video [4]) and so are unsuitable in use cases where exemplar footage is absent. This paper presents a new QVE technique for retrieving video clips using a free-hand *motion-sketch* query, depicting both the *colours* and *trajectories* of objects (Figure 3).

Our focus on colour and motion cues arises from recent work exploring motion-sketch recall of video [7, 6]. These studies report low spatial and temporal accuracy in query sketches, due to both the limited time users are willing to invest constructing a query, and a reliance upon their episodic memory when recalling events [19]. Users typically depict the actions of a few salient objects in a scene, rather than their detailed appearance. Object appearance tends to be depicted using a limited yet approximately correct colour palette — but often using high-level pictograms or canonical shapes that bear little correlation to the object's apparent shape within a real video [7].

The under-complete and approximate nature of sketch has motivated recent "model fitting" approaches to video retrieval; treating the motion-sketch as a space-time model and fitting that model to each clip to evaluate its support and thus the relevance of the clip [6]. We adopt a similar approach, deriving a pattern from the sketched query that is "fitted" to a compact representation of each video. Rather than segmenting the video volume into super-pixels (as in [6, 8]), our video representation comprises a trellis of space-time tokens obtained by tracking and clustering trajectories of visual keypoints. Queries are matched via an efficient Viterbi shortest path search that computes path cost via the Levenshtein distance adapted to operate over our tokens.

1.1 Related Work

Several sketch based retrieval (SBR) algorithms have been proposed within the *image* retrieval domain. Queries are typically colour-blob sketches, deriving descriptors from region topology, global histograms [9], or spectral decomposition [11]. Texture and shape have also been explored [17]. Although such techniques extend to video via key-frame extraction, they do not explicitly accommodate *motion* as a search constraint.

Motion plays an important role in event recall, typically forming a stronger impression upon our episodic memory than visual appearance [19, 7]. However, the combined use of colour and motion cues in sketch based video retrieval (SBVR) has been sparsely researched to date. Trajectory retrieval systems [10, 2, 3, 1] do not consider colour, and typically assume an 'ideal' pre-processor such as global scene segmentation or localised optical flow to extract motion paths. Collomosse *et al* recently proposed a SBVR system in which a linear dynamical system is modelled from the sketch query and fitted to video super-pixels. Unlike previous related approaches (e.g. VideoQ [8]) their system aggregates super-pixels to form sketched objects, and does not assume 'ideal' segmentation of the video into semantic objects. Although this significantly improves precision, the inference step is computationally expensive prohibiting interactive speeds. In this paper we outline a

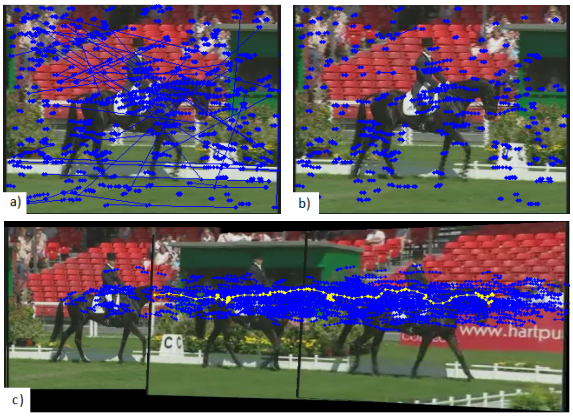

Figure 1. Space-time clustering: (a) initial and (b) filtered correspondences. A GMM component and β-spline motion path (c) consistent with the motion of the horse.

technique for (over-)segmenting video into coherently moving coloured components, and aggregating these to quickly match against a colour-motion description derived from the sketch.

2. Moving Object Description

Upon addition of a new video to the database, we apply shot-detection [20] to segment each video into clips for retrieval. Each clip is parsed as follows.

2.1 Space-time keypoint clustering

Visual keypoints are identified within each frame of the clip using SIFT, and matched with keypoints in the immediately preceding and succeeding frames. Following Lowe *et al.* [14] we iteratively correspond descriptors using the L^1 norm, disregarding correspondences where the distance ratio between the best two matches falls below tolerance. Keypoints detected within TV logos and captions are culled. The spatial position of keypoints are transformed to compensate for any camera motion present in the clip. The camera motion is approximated by estimating the inter-frame homography via MAPSAC using the keypoint correspondences.

The correspondences between camera motion-corrected keypoints result in a set of short-term *keypoint trajectories* (Figure 1a), that we filter to remove erroneous correspondences. Such errors are typically manifested as sporadic, large displacements of keypoints along their trajectory. We assume locally constant motion between adjacent frames, deleting and interpolating the position keypoints whose inter-frame displacement deviates from the local average. Trajectories exhibiting sudden changes in direction are fragmented into separate individual trajectories (Figure 1b). For each trajectory $\mathcal{T}_j$ comprising N_j keypoints $\{(x_j^t, y_j^t); t = \{t_1..t_{N_j}\}\}$ we obtain a feature vector $(v_j^x, v_j^y, p_j^x, p_j^y, f_j)$ where:

$$\bar{v}_j = \frac{1}{N_j - 1} \sum_{t=t_1}^{t_{N_j-1}} (x_j^{t+1} - x_j^t, y_j^{t+1} - y_j^t). \quad (1)$$

$$\bar{p}_j = \frac{1}{N_j} \sum_{t=t_1}^{t_{N_j}} (x_j^t, y_j^t). \quad (2)$$

$$f_j = \frac{1}{t_{N_j}} \sum_{t=t_1}^{t_{N_j}} (t). \quad (3)$$

We perform unsupervised clustering on the space-time trajectories by fitting a Gaussian Mixture Model (GMM) to features of all $\mathcal{T}$, selecting the number of components using a Bayesian MDL criterion [15]. The resulting mixture components correspond to coherently moving objects (or parts thereof) within the clip.

2.2 Extracting motion paths

We extract a representative *motion path* from each clustered component by approximating its global trajectory with a piece-wise cubic β-spline. The solution is unavailable in closed-form due to the typical presence of outliers and piecewise modelling of complex paths. We therefore fit the spline using RANSAC to select a set of control points for the β-spline from the set of keypoints in the corresponding cluster. One keypoint is selected at random from each time instant spanned by cluster, to form the set of control points. The fitness criterion for a putative β-spline is derived from a snake [12] energy term, which we seek to minimize:

$$E = \alpha * E_{int} + \beta * E_{ext} \quad (4)$$

$$E_{int} = \int_{s=0}^{1} \left| d^2 B(s)/ds^2 \right|^2 \quad (5)$$

$$E_{ext} = \Sigma_{t=0}^{T} \left[\frac{1}{|\mathcal{P}_t|} \Sigma_{p \in \mathcal{P}} |p - B(t/T)|^2 \right] \quad (6)$$

where $B(s)$ is the arc-length parameterised β-spline, and $\mathcal{P}_t, t = \{0..T\}$ is the subset of keypoints within the cluster at time t. We set $\alpha = 0.8$, $\beta = 0.2$ to promote smooth fitting of the motion path (Figure 1c).

3. Representation and Retrieval

The clustered components obtained from each clip are processed into a "token" representation (Sec. 3.1), which is stored to facilitate the matching of sketches to clips at query-time (Sec. 3.2).

3.1 Trellis Representation

We discretise each motion path at equal temporal intervals and record the motion vectors between points on the path at adjacent intervals. The number of intervals I (10) is constant across all clips. We augment each intervals' motion vector with a colour histogram sampled from the corresponding video frame. Quantization is first performed over the whole clip to determine the set of (32) colour bins for the histogram. The histogram for each interval on the motion path is populated by sampling the colour of the video frame beneath each keypoint within that temporal interval. To mitigate against video noise when point-sampling colour, we segment the video frame into coloured regions using mean-shift [5]. The colour of keypoint is deemed to be the (quantized) mean colour of the underlying region, and a weighted contribution is made to the histogram proportional to the area of that region. Once constructed, the histogram is normalised to represent the relative colour distribution of the moving object.

Having processed all clustered motion components within the clip, we construct a directed acyclic graph $\mathcal{V}$ from the tokens. Each node $\mathcal{V}_{mt}$ corresponds to the token from the m^{th} motion path at time interval $t \in 1..I$. It is likely multiple tokens will co-exist at a given time, as many objects may exist at a given instant and objects themselves may be over-segmented. Edges are created from $\mathcal{V}_{mt}$ to $\mathcal{V}_{\hat{m}\hat{t}} \forall \hat{m}$ at $\hat{t}$ (where $\hat{t}$ is the next time interval containing at least one token). This produces a fully-connected "trellis" (Figure 2), where each vertical layer contains zero or more tokens. We connect a *source* node to $\mathcal{V}_{m1}$ and a *sink* from $\mathcal{V}_{mI}$.

3.2 Matching Tokens

Our system accepts query motion sketches in the form of a temporally ordered list of strokes. A pictogram recognition algorithm [7] is applied to locate motion cues (arrows) within the sketch. Remaining strokes are grouped into objects using graph-cut [7]. Objects are associated with motion cues by proximity.

Motion sketches are temporally ambiguous; trajectories indicate the sequence of direction changes along an object's motion path, but contain no information relating to speed or acceleration. We assume a constant speed along the sketched trajectory, but observe that direction changes on the path are inaccurately depicted with respect to time. Our matching process adopts an edit distance metric to accommodate this variation.

Given a query sketch, we first obtain a sequence of tokens $\mathcal{Q} = \{Q_1, ..., Q_I\}$ from the trajectory of each sketched object. As in Sec. 3.1, the trajectory is divided into I intervals of equal arc-length. Each interval yields a token comprising a motion direction and colour histogram. Sketched objects share a histogram obtained from strokes comprising the object.

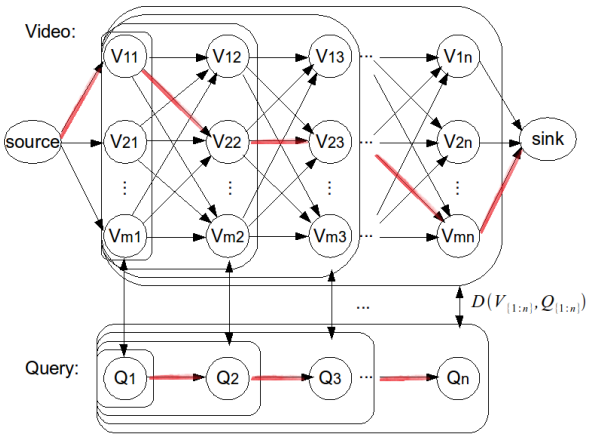

Figure 2. Motion paths are discretised in time to form motion-colour tokens. Tokens are arranged into a trellis and Viterbi shortest path computed across the graph. Past cost (red) is the edit distance between tokens on the path, and a token sequence derived from the query.

Our matching strategy is to evaluate the support that each clip (trellis of tokens V_{mt}) holds for the sketch (tokens $\mathcal{Q}_{1...I}$); clips are ranked according to this support. We compute the Viterbi shortest path from source to sink, computing path cost as the edit distance between tokens on the putative shortest path and those in $\mathcal{Q}$ (Figure 2). The path cost $C_t(\mathcal{Q}, \mathcal{V})$ (from source to time t), and the corresponding token sequence $\alpha_t \subseteq V_{mt}$ are:

$$\begin{aligned} \alpha_t &= \{\alpha_{t-1}\ , \\ &\quad \underset{m}{\text{argmin}}\ D\left(\mathcal{Q}_{1...t}, \{\alpha_{t-1}, \mathcal{V}_{mt}\}\right)\} \\ C_t(\mathcal{Q}, \mathcal{V}) &= C_{t-1}(\mathcal{Q}, \mathcal{V}) + \log D(\mathcal{Q}_{1...t}\ , \alpha_t) \quad (7) \end{aligned}$$

where $C_0(Q, V) = 1$, $\alpha_0 = \emptyset$ and $D(q, v)$ expresses the (dis-)similarity of two token sequences as an edit distance. $D(q, v)$ is computable using Levenshtein's algorithm [13] in $O(n^2)$, where $n = min(|q|, |v|)$ i.e. $n = I$ bounds n in our system. This is the minimal 'cost' of operations required to transform q into v. Operations comprise token insertions, deletions and substitutions. The substitution cost between two tokens $(a \in q, b \in v)$ is a user weighted product of the colour similarity $D_c(.)$ and the motion similarity $M_c(.)$:

$$\begin{aligned} D_c(a, b) &= -\sum_{i=1}^{32} \sum_{j=1}^{|b|} C_a(i) C_b(j) (1 - |\mathcal{L}_a(i) - \mathcal{L}_b(j)|) \\ D_m(a, b) &= -\frac{M_a \cdot M_b}{|M_a||M_b|} \quad (8) \end{aligned}$$

where $C_{a,b}(i)$ is the i^{th} colour histogram bin, $\mathcal{L}_{a,b}(.)$ the corresponding normalised CIELab colour, and $M_{a,b}$ is the motion vector of individual tokens a and b. Insertion and deletion costs are functions of D_m only.

4. Evaluation and Discussion

We evaluated our system using five datasets: (i) a subset of the public KTH activity dataset [16], comprising 200 clips of people walking/running; (ii) 120 horse racing clips (Horse); (iii) 29 motor racing (Car); and (iv) 83 diving clips from the 2008 Olympic Games (Diving). We also ran experiments on a combined dataset (HCD) containing 232 clips of 'Horse', 'Car' and 'Diving'.

Figure 3 (left) plots the precision-recall curves for all dataset. The Mean Average Precision (MAP) scores for a set of 17 queries over each set were Horse (54.4%), Car (55.0%), KTH (38.8%), Diving (30.0%), HCD (28.2%). Queries comprised both simple linear and more complex trajectories (Fig. 3, right). In the more challenging examples of motion (e.g. diving), the trajectory clustering process fragmented objects into several space-time volumes. However our matching process correctly aggregated the respective tokens to retrieve the correct object. Our lower MAP over the KTH dataset is due to absence of colour cues, which can cause the shortest path to visit erroneous regions. Average running time for a match was $20-30$ms per clip on unoptimized C code (i.e. a few seconds per query).

This work demonstrates that sketch based QVE for video retrieval is viable using sketched depictions containing only colour and motion cues. Furthermore it demonstrates that a part aggregation/model-fitting approach to SBVR can accommodate non-trivial motion trajectories, and achieve interactive speeds. Improvements focus on improved robustness to clutter and introducing relevance feedback to interactively learn the weights between colour and motion in eq. (8).

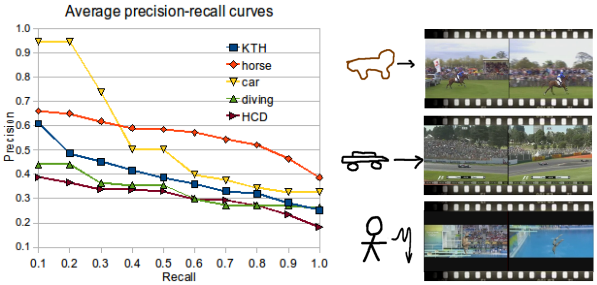

Figure 3. Precision-recall curves for the 5 datasets (left). Example queries and the top clip for Horse, Car and Diving (right).

References

[1] N. Anjum and A. Cavallaro. Multifeature object trajectory clustering for video analysis. *IEEE Trans. on Circuits and Systems for Video*, 18(11):1555–1564, 2008.

[2] G. Antonini and J. P. Thiran. Counting pedestrians in video sequences using trajectory clustering. *IEEE Tran. on Circuits and Systems for Video*, 16(8):1008–1020, 2006.

[3] F. I. Bashir, A. A. Khokhar, and D. Schonfeld. Real-time motion trajectory-based indexing and retrieval of video sequences. *IEEE Trans. Multimedia*, 9(1):58–65, Jan. 2007.

[4] M. Bertini, A. Del Bimbo, and W. Nunziati. Video clip matching using mpeg-7 descriptors and edit distance. In *CIVR*, pages 133–142, July 2006.

[5] C. M. Christoudias, B. Georgescu, and P. Meer. Synergism in low level vision. *ICPR*, 4:40150, 2002.

[6] J. Collomosse, G. Mcneill, and Y. Qian. Storyboard sketches for content based video retrieval. In *ICCV*, 2009.

[7] J. Collomosse, G. Mcneill, and L. Watts. Free-hand sketch grouping for video retrieval. In *ICPR*, 2008.

[8] S. fu Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong. VideoQ: An automated content based video search system using visual cues. In *ACM Multimedia*, pages 313–324, 1997.

[9] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Effcient color histogram indexing for quadratic distance. *IEEE PAMI*, 17(7):729–736, 1995.

[10] J. Hsieh, S. Yu, and Y. Chen. Motion-based video retrieval by trajectory matching. *IEEE Tran. on Circuits and Systems for Video*, 16(3):396–409, 2006.

[11] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multi-resolution image querying. In *Proc. ACM SIGGRAPH*, pages 277–286, Aug. 1995.

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, V1(4):321–331, 1988.

[13] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, Soviet Physics Doklady, 1966.

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[15] S. Roberts, D. Husmeier, R. Penny, and l. Rezek. Bayesian approaches to gaussian mixture modeling. *IEEE Trans. on PAMI*, 20(11):1133–1142, 1998.

[16] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. *ICPR*, 3:32–36, 2004.

[17] E. D. Sciascio, G. Mingolla, and M. Mongiello. CBIR over the web using query by sketch and relevance feedback. In *Proc. Intl.Conf. VISUAL*, pages 123–130, June 1999.

[18] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *ICCV*, pages 2:1470–1477, 2003.

[19] E. Tulving. *Elements of episodic memory*. Oxford Clarendon, 1983. ISBN: 0-198-521251.

[20] H. Zhang, A. Kankanhalli, and S. W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1(1):10–28, 1993.