A Sketch-Based Approach for Detecting Common Human Actions

Evan Suma* Justin Babbs Richard Souvenir

University of North Carolina at Charlotte

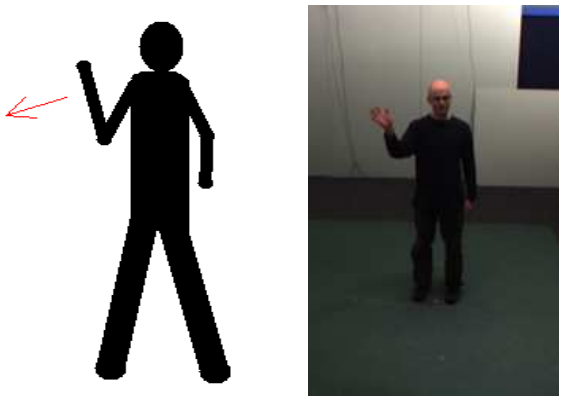
Abstract

Many application domains, including athletics and surveillance, can benefit from quickly detecting specific actions in video. We present a method for detecting common human action in these video streams using intuitive sketches of objects and motion cues. The application presented in this paper is an automated end-to-end system which (1) interprets the sketch input, (2) generates a query video based on motion cues, and (3) incorporates a recently developed video similarity measure for matching. Our preliminary results show videos that correspond strongly to the search concept generally score higher than unrelated videos.

1. Introduction

Automated human activity detection from video is an important problem that plays a role in many domains, including athletics and surveillance. Possible applications include searching archived athletic footage for an instance of a particular action or detecting instances of a particular action in a real-time security setting. However, searching through a video (or a database of videos) is still an open, challenging problem. In a typical video querying scenario, a user needs to provide some type of description of the intended action. Commercial solutions such as Google Video and Youtube typically employ search methods which do not match directly to the content of the video; instead, a textual query is matched to metadata associated with the video such as the title, description, or user comments. The possibility of incomplete or incorrect metadata is a well-known limitation to this approach. This has led to a host of methods that fall under the umbrella of content-based video retrieval (CBVR).

The literature on CBVR is extensive; see [1] and [2] for surveys. Multiple taxonomies exist for the classification of CBVR approaches. To ground the method we present in this paper, we focus on two broad classes of techniques based on the query input: (1) text-based (or concept-based) approaches and (2) example-based approaches. Text-based approaches, such as [3], typically rely on some (semi-supervised or unsupervised) step of grouping videos together based on some concept and refining the search within each cluster to obtain the desired result. Example-based approaches, as done in [4], typically match features of the query video against those in the database and return high-scoring matches.



(a) (b)

Figure 1: (a) An example input sketch of a wave action. (b) A frame from a high-scoring video search result.

There are advantages and disadvantages to both methods. Text-based approaches work when the content of the videos can be described succinctly. Also, fine-tuning search results using new queries is simple since the user can select new keywords to try. However, these methods fail in the cases where the query is ambiguous (e.g., “driving” for cars versus swinging a golf club). Additionally, videos may contain many secondary objects that may be overlooked during grouping if classification is based upon the primary object or action in the video.

Example-based approaches can overcome the limitation of ambiguous searches because a query video is more informative than a text label and many approaches exist for matching pairs of videos (feature-based, statistics-based, etc.). However, finding representative videos to use for querying other videos can be difficult. More specifically, if a video strongly matching a search concept

* email: {easuma, jlbabbs, souvenir}@uncc.edu

were easily obtainable, it might not be necessary to perform the query in the first place. Thus, such approaches are limited to scenarios in which the user wishes to find other similar videos to one that is already available.

In this paper, we present a method for querying databases of videos containing articulated objects, such as moving humans. Our method relies on using intuitive hand-drawn sketches of objects and motion cues as queries for video retrieval. We believe that allowing the user to provide the input in this manner combines the best features of both the text- and example-based approaches. The application presented in this paper is an automated end-to-end system which (1) interprets the sketch input, (2) generates a query video based on motion cues, and (3) incorporates a recently developed video similarity measure for matching. The resulting search query is more informative than a text-based approach and does not require that the user explicitly provide an example video. Figure 1 shows an example input sketch and a keyframe from a high-scoring video search result.

Sketch recognition is a related area where the goal is to infer the semantics of an input sketch. Unlike those methods, (e.g. [5]), we are not interested in classifying the action represented in the sketch, nor do we need to collect the gesture information associated with creating the sketch. Our goal is to search a video database or stream for a conceptual match. Searching image and video databases using sketches is a natural solution to the textual input problem that has previously been explored. In [6], the author presents a method using a sketch-based system to search a large static image database. In [7], the authors present a system which queries videos using sketches of motion cues and mainly provides for queries focusing on the translation of objects in a viewing frame and not the finer-grained articulated motions that our system is capable of matching.

The paper is organized as follows. In Section 2, we describe our approach for interpreting sketches. In Section 3, we explain our method for matching a query video against a database. In Section 4, we show the preliminary results of testing our method using a publicly available human motion data set. In Section 5, we discuss some potential uses for the ideas presented here and the limitations of our current approach. In Section 6, we conclude and discuss future directions for this work.

2. Interpreting sketches

The input image can either be provided as a freehand sketch or generated from a human body model using a point-and-click application. While the former allows for the motion of arbitrary objects, the latter is more robust for human motion because the skeletal structure is known in advance. In order to search a video database based on an input sketch, it is first necessary to examine the motion cues and infer what motion is being described. We selected arrows as an intuitive method of depicting moving objects. For both types of input, the arrows are used to infer the intended motion of image components, and an animation sequence is generated from the input image. Subsequently, the resulting video is used as a template for an example-based search of a video database. In this section, we describe how we interpret sketches.

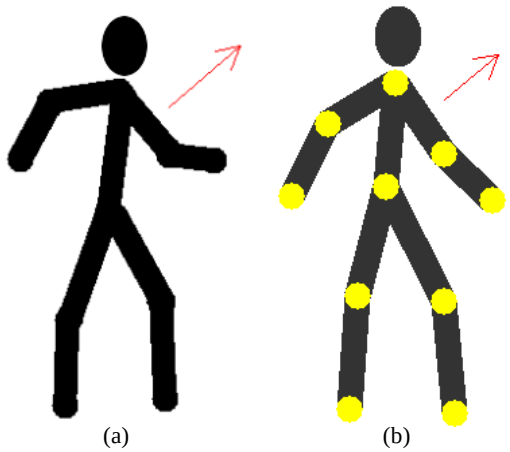


Figure 2: (a) A sketch generated using the freehand drawing interface. (b) A sketch generated using the human body model interface. The figure is manipulated by clicking and dragging at the joint locations.

2.1. Input methods

The user has the capability to initialize a search query by supplying a freehand sketch (see Figure 2.a). Our system includes a sketch drawing application which provides two drawing modes: (1) figure mode and (2) arrow mode. In figure mode, the user has access to various drawing tools to specify the structure of the figure. In arrow mode, the user can click and drag on the image to draw a motion arrow.

For representing human motion, we also a provide point-and-click interface to produce sketches by manipulating a human body model (see Figure 2.b). This interface works similarly to the sketch drawing application, except the user does not explicitly draw the figure. Instead, limbs on a human puppet figure are positioned and resized by clicking and dragging the joint locations. This method is a more robust approach than freehand sketching because the connectivity of the skeleton, as well as the locations of joints, is already known to the system, which reduces the errors associated with misinterpretation of the sketch.

2.2. Interpreting arrows

While we are primarily concerned with videos involving human motion, a freehand sketch could potentially contain objects other than human figures. Furthermore, an arrow in these images can either refer to a specific moving component of an object (for example, a limb on a human) or the entire object moving as a whole (see Figure 3.a). Thus, it is necessary to examine the object's skeleton to determine the joint locations which separate the movable components of the figure. To accomplish this, a freehand sketch is first blurred using a Gaussian kernel and thresholded to produce a binary image. Then, a medial axis transformation algorithm is applied to generate a skeleton of the image.

We project the arrow in the reverse direction on the skeleton image until the skeleton is contacted (see Figure 3.b). We assume that the point of contact, which we call the origination point, lies on the moving component being referenced by the arrow. To determine the component each origination point lies on, the Hough transform [8] is applied on a local neighborhood around the point. In our implementation, the local neighborhood is defined by tracing n_l pixels along the skeleton from the origination point in the direction towards the center of mass. In our tests, we found that 20 pixels accurately sample the line segment in a 250 x 300 pixel sketch. The largest peak in the Hough transform is considered to be the line representing the component's skeleton. The endpoints of the line segment that corresponds to this line in the image are then calculated. The endpoint that is closest along the skeleton towards the center of mass is the detected joint location for the component (see Figure 3.c). If the center of mass is reached before the end of the line segment, then we interpret this to mean that the arrow motion most likely corresponds to a movement of the overall object.

For sketches produced by manipulating the human body model, interpreting the arrows is a much simpler process since the connectivity of the skeleton and locations of joints are already known in advance. The origination points are calculated in a similar manner by projecting each arrow in reverse until the skeleton is contacted.

2.3. Sketch animation

To generate a sequence of images which estimates the motion represented by a sketch, it is first necessary to segment the image into separate components at the joint locations. This is done so that individual components can be translated and rotated according to the motion indicated by the arrows. At each joint, the image is 'cut' along the normal to the skeleton, separating the component from the rest of the object. In complex skeletons such as human figures, components may be the children of other components (such as the forearm and upper arm). To account for this possibility, the parent of each component, if any, is calculated by tracing from the joint location along the skeleton towards the center of mass to detect other joints. For images generated from a human body model, the parent-child relationship between joints is already known. Additionally, it is not necessary to segment the image into separate components, since the system can generate an image with transformed components from the ground up.

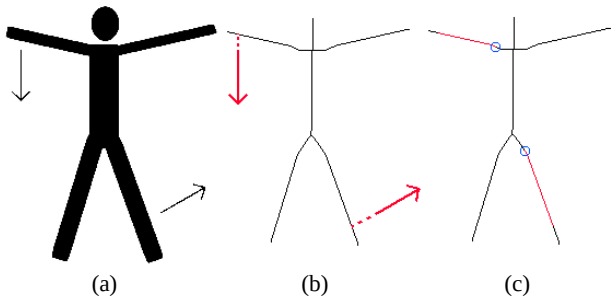


Figure 3: (a) A freehand sketch with arrows indicating movement. (b) The image after skeletonization with arrows projected back onto the skeleton. (c) The detected line segments and joint locations.

In the rotation phase, the overall angle of component rotation is determined by computing the angular difference between the component vector and the vector formed by the joint location and arrow end point. For each frame, each component is rotated by n_r degrees, where n_r is total angular rotation divided by the number of frames. Child components are rotated first, and added to the parent for rotation. Thus, motions from multiple components are combined to produce more complex motion sequences (see Figure 4). Though the motion sequence may not be visually pleasing, the imperfections in the produced video will not significantly affect the matching process, as we demonstrate in the next section.

3. Matching videos

We chose a feature-based approach to matching videos. For this problem, we need a representation that models the content of the video rather than the appearance, since sketches typically do not share appearance characteristics with real video.

3.1. Feature descriptor

We implement the self-similarity descriptor presented in [9], which measures self-similarity as a local image property and has several advantages over other techniques. Despite large photometric differences between two images of similar objects, the corresponding self-similarity descriptors are very similar. It performs well in matching similar objects transformed by locally affine or non-rigid

Figure 4: (a) A freehand input sketch with multiple arrows and frames from the generated video sequence. (b) A sketch generated from a human body model and the corresponding video sequence.

deformations, which makes it robust to inexact sketches or motion extrapolation.

This descriptor provides a compact description of the neighborhood around a given pixel. It is a multi-dimensional spatial histogram (3D, in our case) which represents how similar the neighboring pixel regions are to a given image location. Figure 5 shows a descriptor for 2 matching regions in a sketch and an image. Even though the appearance of each image differs, the content of both regions are similar and, thus, the histogram representations match. For video sequences, the self-similarity descriptor is extended into space-time to produce a correlation volume. However, for clarity, we only show the 2D representation.

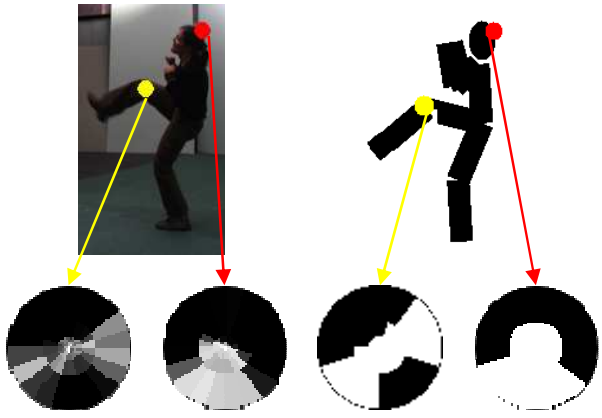


Figure 5: Self-similarity descriptors for two pixels in a sample frame and sketch. The self-similarity of the regions surrounding each location is similar and, thus, the histogram representations are close.

This feature can be calculated at every pixel location in a video. However, matching two videos only requires a small number of corresponding feature matches, so we compute a self-similarity descriptor at n_d locations within the silhouette, selected at random, where ($n_d \leq n$) and n is the number of pixel locations within the silhouette. This method is significantly faster because it only examines pixels that are determined to be informative, based on their presence on the silhouette of the object. The resulting set of descriptors is combined to form an ensemble which describes the spatio-temporal self-similarity of the entire video. When performing searches on a collection of videos, the feature vectors can be computed offline and stored in a database for quicker searching. Thus, during retrieval, it is only necessary to compute the ensemble for the input sketch.

3.2. Matching animated sketches with videos

Matching is accomplished in a manner similar to [10], which is based on the Generalized Hough Transform [11]. Figure 6 illustrates the process. For the animated sketch video, we calculate n_d self-similarity features. Figure 6.a depicts the locations of 4 example features. For each feature in the animated sketch, we calculate the displacement(s) from a location that we call the center of action. The result of this is a codebook which provides list of displacements given a feature (see Figure 6.b). Using this codebook, we now score all of the locations in the videos of our database. For each feature from the database video, we perform a lookup using the codebook and consider matches to all features within ε units. This casts a vote (in the space of the database video) for the center of

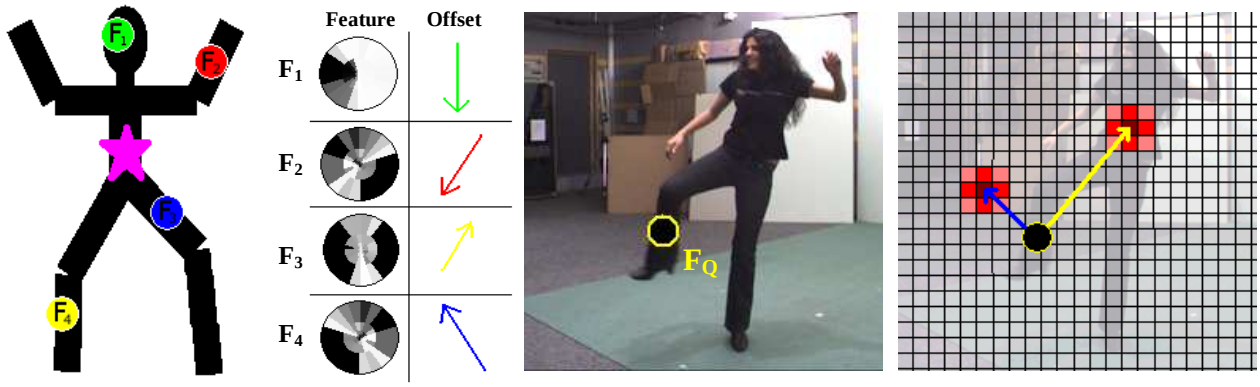


Figure 6: Illustration of the matching process using the Generalized Hough Transform. (a) F_1 - F_4 show the location of 4 example features calculated from the sketch video. The star denotes the center of action. (b) The codebook contains the offset vectors for each feature to the center of action. (c) F_Q represents one (of many) features calculated from the test video. (d) For all the entries in the codebook within some threshold distance, ε, of F_Q (F_3 and F_4, in this case), we cast votes for potential the center of action using those offsets and the location of F_Q. This process is repeated for all of the features in the test video to find the areas with high accumulated votes.

action (see Figures 6.c and 6.d). For robustness, we employ a common trick where each detected feature casts votes for a region (represented with a Gaussian) rather than a single pixel location. This procedure is followed for all of the features in the database video and the votes are accumulated. For each video in the database, we compare the number of votes for the highest peak to some pre-defined threshold, t, to determine if the video was a match to the input sketch.

4. Results

We used the Inria XMAS Motion Acquisition Sequences (IXMAS) dataset [12] to produce the results in this section. This data was collected by 5 calibrated, synchronized cameras and contained various actors performing 13 different actions. We tested the system using three sketches intended to represent kick, wave, and throw actions. Our database contained examples of various actors performing these and other common human actions from typical camera viewing angles. For each input sketch, we calculated the matching scores to all of the videos in the database. Figure 7 shows sample results for each query. Each row shows the input sketch and a keyframe of one of the videos from the database. For each example, we show a keyframe of the highest scoring video and 2 other (low-scoring) examples. In all of our tests, the intended video was one of the top-scoring matches to our input sketches.

5. Discussion

Sketches are an intuitive way of providing input, but they can be flawed, or, in the worst case, misrepresentative if they do not properly portray the action that the user intends. Unlike text-based approaches, which only exhibit variability in the output, this method can introduce variation in the input. So, the best strategy for interpreting the results of this work is not straightforward because it is not clear how to quantitatively assess how well a sketch represents a specific action. Therefore, analyzing the accuracy of the output is particularly difficult when the accuracy of the input is not known.

The method introduced in this paper represents a 3D motion with a 2D sketch. There are inherent limitations to this approach. While a sketch can represent motion that varies with multiple degrees of freedom, this method is limited to succinct, atomic actions and restricted in the viewpoint. It may be possible to overcome some of these limitations by developing more sophisticated 2D representations for common human actions. While such an approach may be less intuitive, it might allow greater versatility in the types of actions that can be represented in a sketch.

6. Conclusions and Future Work

We presented a method for the detection of human actions based on input from a sketch. While there are inherent limitations in using 2D sketches to search for 3D data, we feel that this approach can still be useful to domains such as athletics and surveillance where the large quantities of video data contain examples which can be succinctly described by sketches with motion cues. Future directions include using tree-based approaches, such as [13], for faster matching to a database or real-time video feeds.

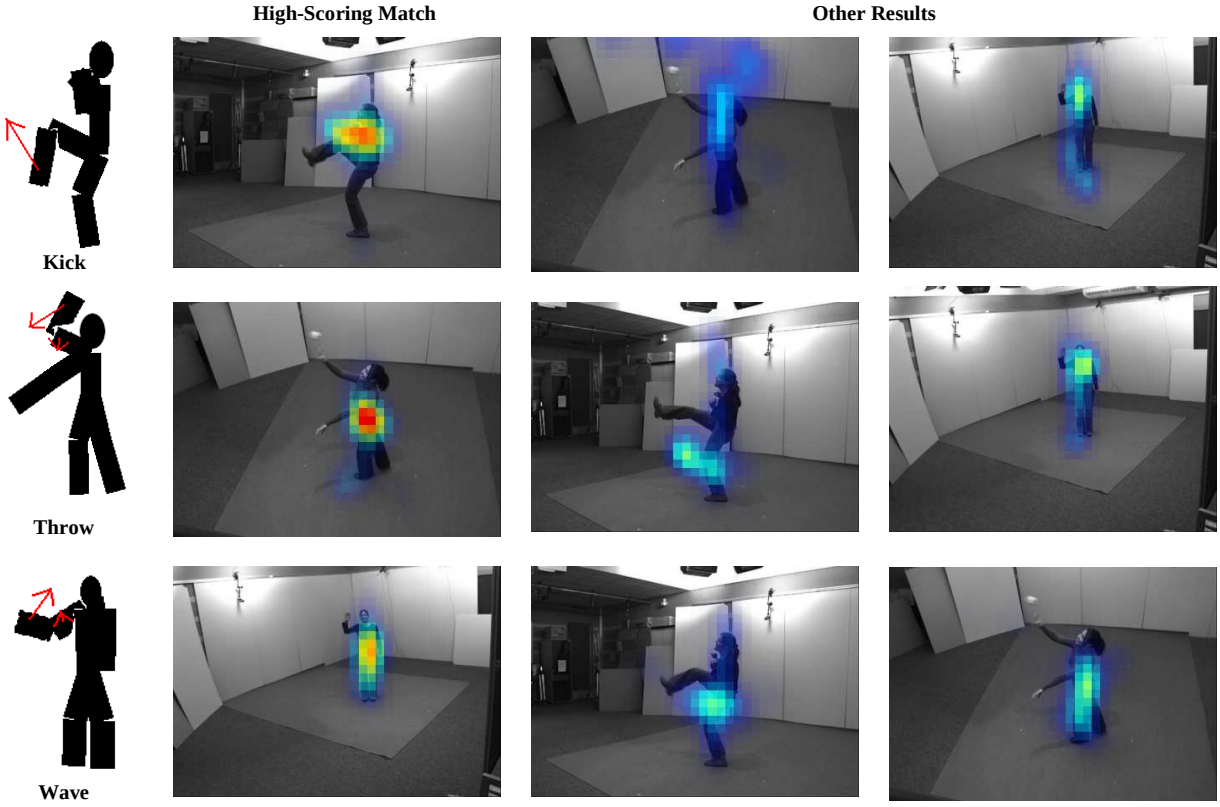


Figure 7: Results of comparing 3 input sketches to IXMAS videos. Each row shows the input sketch and a representative keyframe from a database video. The color of the overlay represents the total votes for the center of the query action.

References

[1] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 2, pp. 1-19, 2006.

[2] S. Marchand-Maillet. Content-Based Video Retrieval: An Overview. Tech. Rep. 00.06, CUI – University of Geneva, Geneva, Switzerland, 2000.

[3] M.R. Naphade and T.S. Huang. Semantic video indexing using a probabilistic framework. 15th International Conference on Pattern Recognition, vol. 3, pp. 79-84, 2000.

[4] C. Taskiran, J. Chen, A. Albiol, L. Torres, C.A. Bouman, and E.J. Delp. ViBE: A compressed video database structured for active browsing and search. IEEE Transactions on Multimedia, vol. 6(1), pp. 103-118, 2004.

[5] B. Paulson and T. Hammond. MARQS: Retrieving Sketches Using Domain- and Style-Independent Features Learned from a Single Example Using a Dual-Classifier. IWUMUI 2007, 2007.

[6] M.S. Lew. Next-generation Web searches for visual content. Computer, vol. 33-11, pp. 46-53, 2000.

[7] S. Chang, W. Chen, H.J. Meng, H. Sundaram, and D. Zhong. VideoQ: an automated content based video search system using visual cues. Fifth ACM International Conference on Multimedia, pp. 313-324, 1997.

[8] R.O. Duda and P.E. Hart. Use of the Hough transformation to detect lines and curves in pictures. Communications of the ACM, vol. 15(1), pp. 11-15, 1972.

[9] E. Shechtman and M. Irani. Matching Local Self-Similarities across Images and Videos. IEEE CVPR 2007, 2007.

[10] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. ECCV'04 Workshop on Statistical Learning in Computer Vision, pp. 17-32, 2004.

[11] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, pp. 714-725, 1987.

[12] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. Computer Vision and Image Understanding, vol 104(2), pp. 249-257, 2006.

[13] D. Nister and H. Stewenius. Scalable Recognition with a Vocabulary Tree. IEEE CVPR 2006, vol. 2, pp. 2161-2168, 2006.