Recognizing action events from multiple viewpoints

Tanveer Syeda-Mahmood  A. Vasilescu
IBM Almaden Research Center  Dept. of Computer Science
650 Harry Road, San Jose 95120  University of Toronto, Canada.
stf@almaden.ibm.com  alemov@hotmail.com
S. Sethi
Dept. of Computer Science
Boston University
ssethi@bu.edu

Abstract

A first step towards an understanding of the semantic content in a video is the reliable detection and recognition of actions performed by objects. This is a difficult problem due to the enormous variability in an action’s appearance when seen from different viewpoints and/or at different times. In this paper we address the recognition of actions by taking a novel approach that models actions as special types of 3d objects. Specifically, we observe that any action can be represented as a generalized cylinder, called the action cylinder. Reliable recognition is achieved by recovering the viewpoint transformation between reference (model) and given action cylinders. A set of 8 corresponding points from time-wise corresponding cross-sections is shown to be sufficient to align the two cylinders under perspective projection. A surprising conclusion from visualizing actions as objects is that rigid, articulated, and non-rigid actions can all be modeled in a uniform framework.

1 Introduction

Action events are a common occurrence in many real world applications such as surveillance, human-computer interfaces, interactive environments and content-based retrieval. For example, in surveillance, both usual and unusual activities may be useful to monitor. In distributed learning, the emphasis on a topic can be learned from the pointing actions made by the speaker. Automatic interpretation of sign language requires the analysis of actions. Finally, observing user actions or gestures is important in human-computer interaction and the design of HCI systems.

Despite the progress made in image and video content analysis, detecting and recognizing action events has remained an outstanding problem due to several reasons. First, since each action is associated with an object, it would require object detection. Secondly, the segmentation of actions continuously performed by the object into distinct actions, poses challenging problems. Finally, an action can show enormous variability in image appearance when seen at different time intervals. Changes in imaging conditions such as illumination changes, viewpoint changes, and pure variations in execution style can considerably alter the appearance of an action in an image from its original description. Finally, an action may only be partially executed or partially visible in image during its execution in space, making its automatic detection and recognition difficult.

Although the problem of action recognition has been actively addressed, particularly in the area of gesture recognition, the above problems still remain outstanding. Existing work in this area can be broadly classified into region-based, trajectory-based, and part-based approaches. The region-based approaches analyze spatio-temporal patterns of actions. These can be two-dimensional shapes, as in the case of temporal templates[7], view models[6] or three-dimensional shapes as in the case of XYT cube [14] or image solid[15]. The trajectory-based approaches analyze temporal trajectories of features for properties such as velocity, direction and speed, joint angles, and spatio-temporal curvature of the temporal trajectories to classify actions [13, 17]. More complex spatial models of human action treat the body as a set of connected parts[2, 4, 10]. There has also been work on the recognition of activities using the temporal evolution of the parameters of such models[1, 8].

From our understanding of work in this area, all approaches to action recognition are sensitive to changes in viewpoint. For example, it is common knowledge
that the recognition accuracy can drop considerably for actions such as sitting actions seen sideways and frontally. Further, 3d spatio-temporal shapes such as XYT cube[14] and image solid[15] being formed from the entire image sequence, are useful for detecting high-level patterns such as those in periodic actions. In addition, the trajectory-based approaches require accurate tracking which is often difficult for people due to clothing and self-occlusions. While tolerance to variations such as local changes of speed are not explicitly addressed, most approaches do take some of the execution style differences into account during matching, using techniques such as dynamic time warping [6]. Finally, not many approaches have addressed the recognition of actions from their partial appearance.

In this paper, we address the problem of robust recognition of action events. Specifically, we focus on the recognition of an action performed by an object when seen from a different viewpoint. In particular, we observe that the shape formed from the successive perspective projections of an object executing an action can be visualized as a generalized cylinder, called the action cylinder. Using this representation, action recognition involves finding a set of corresponding features between model and given action cylinders and verifying the correspondence through projection of the model cylinder.

The paper makes several novel contributions to advance the state of the art in action recognition. First, by expressing actions as shapes, it allows the entire machinery of object recognition techniques to be brought to bear on the action recognition problem. It also allows all complex action types, rigid, articulated or non-rigid, to be modeled in a uniform framework. Next, it shows that a set of eight corresponding features from corresponding cross-sections are sufficient to recover the viewpoint transformation as well as the time correspondence needed for recognizing the action. Finally, by combining the effect of shape and motion in a 3d shape, it allows their relative contributions to be explicitly modeled.

The rest of the paper is organized as follows. In Section 2, we model actions as shapes through action cylinders. In Section 3, we discuss the recognition of actions using action cylinders. In Section 4, we present our action recognition algorithm. Finally, in Section 5, we present results on the performance of our algorithm on a large action data set.

2 Modeling actions as shapes

For purposes of this paper, we define an action to be the motion sequence executed by a 3D object. We now analyze the shape formed by an action using the following assumptions.

Assumptions: 1) The imaging situation depicts a single object undergoing an action. 2) It is viewed using a single fixed camera under full perspective projection. 3) The object can be separated from the background by some means (several methods of background subtraction are available[8]), and finally, 4) a single action has been segmented from other previous and next actions done by an object. The latter is critical to the identity of an action as an object.

Under the above assumptions, as an object undergoes motion, a point P(t) = (X(t), Y(t), Z(t)) on the 3d object at time instant t is projected into the image at location p(t) = (x(t), y(t)). If we consider the spatio-temporal solid formed from the time varying projections of object features (x(t), y(t)) as a function of time t, the resulting 3D shape formed in image-time space is a cylinder, which we call the action cylinder.

Observation: The successive perspective projections of every isolated action of a 3d object moving in space can be represented as a generalized cylinder in image-time space.

As shown in Figure 1, a generalized cylinder is represented by (A(s), E(s, θ, α), Φ(s)) where A(s) is the axis of revolution of the cylinder parameterized by s. θ, α specify the cross-section in the cylinder's coordinate frame, and Φ(s) is the change in orientation of the cross-sections with respect to the axis of revolution. Action cylinders are generalized cylinders by the following equivalence. E(s, θ, α) represents the 2d projections of the 3d object as it undergoes motion and is clearly affected by the shape of the object. The change in the cross-sections over time is due to motion as well as effects of perspective projection. Φ(s) represents the change in the apparent orientation of the 2d shapes (projections) due to both motion and effect
of projection. Finally, \( A(s) \) is the axis of revolution that generated the resulting shape from the successive cross-sections and is affected by shape, motion, and projection.

Thus the action cylinder is a representation for capturing the combined shape and motion information of an object. Because it is formed from the projections, it also accounts for the effect of projection. Other variations of the action cylinder are also possible, such as a feature-only version using corners within bounding and interior contours of the cross-sections. In this paper, we use a feature-based representation of the action cylinder in all our analysis.

**Action cylinder vs. other action representations**

It is interesting to compare and contrast the action cylinder with other action representations. Action cylinders are similar to temporal templates [7] and XYT cubes [14, 15] in that they are all region-based descriptors. Unlike the temporal templates, action cylinders are 3D shapes that preserve the order with respect to time. Unlike other 3D spatio-temporal shapes that form cubes from the entire image, the action cylinder represents only the object undergoing motion thus capturing the action in a precise fashion. Also, in other spatio-temporal approaches, the recognition of the action is achieved by extracting and classifying attributes. In contrast, we perform a direct geometric match of the entire shape of the action cylinders for recognizing actions. Since action cylinders are unique for a particular action and a specific object, they can distinguish the same action done by a different object unlike temporal trajectories [13, 17]. The temporal trajectories of different points on an object can appear quite different and may not represent the action unless the entire 3D image-time shape was considered. Further, action cylinders allow detailed capture of the action without requiring complex tracking of the entire object. When they are represented using corner features, action cylinders are relative insensitive to illumination changes. The action cylinders can also be sub-sampled for slow actions without affecting the overall perception of the cylinder’s shape and hence the action. Finally, because of the built-in redundancy in an action cylinder due to appearing and disappearing features from the same object location, they are relatively insensitive to feature detection errors.

3 Recognizing actions using action cylinders

Using the action cylinder representation, the problem of action recognition can now be posed as a model-based object recognition problem. That is, the problem is to determine if a given action cylinder of an isolated action matches one of a set of stored action cylinders (called model cylinders). To address the matching in a manner analogous to model-based object recognition, we need to take into account the effect of various imaging conditions such as illumination changes, viewpoint changes, occlusions and noise, that can cause the two cylinders to appear different. In addition, since the action cylinders represent motion information, we need to consider differences produced due to changes in execution style that include velocity and acceleration variations. We now describe a way to match two action cylinders that address some of the above issues. Specifically, we focus on recognizing actions when seen from multiple viewpoints.

3.1 Recognizing actions under changes in viewpoint

With cross-sections coming from perspective projections of the 3D object, an action cylinder is clearly not viewpoint invariant. It is, however, possible to recover the viewpoint transformation by a match of feature points across the two action cylinders. To see this, consider an action cylinder \( \hat{S}(t) \) seen from one viewpoint \( V_1 \). Let the stored representation be of the same action but seen from a different viewpoint \( V_2 \) and denoted by the model cylinder \( S(t) \). Using a well-known relation between two views of an object seen from viewpoint \( V_2 \) at the same time instant as the 2D projection \( \hat{S}(t) \), in general, the time profiles \( t_0 \) and \( t \) are identical only in the special case when the action is viewed simultaneously at the same frame rate.

To recognize an action under changes in viewpoint, we use a well-known relation between two views of an object under perspective projection derived in several papers including [16]. We incorporate time information into the notation to specifically address the shapes induced by actions.

Using matrix notation, and choosing the viewpoint of the model cylinder as the reference viewpoint, a point on the 3D object \( P(t_0) = (X(t_0), Y(t_0), Z(t_0))^T \) projects to point \( p(t_0) = (x(t_0), y(t_0))^T \) in \( S(t_0) \) and \( \hat{p}(t_0) = (\hat{x}(t_0), \hat{y}(t_0))^T \) in \( \hat{S}(t_0) \). Using homogeneous coordinates, \( \hat{P}(t_0) = (X(t_0), Y(t_0), Z(t_0), 1)^T \), \( \hat{p}(t_0) = (x(t_0), y(t_0))^T \) and \( \hat{p}(t_0) = (\hat{x}(t_0), \hat{y}(t_0))^T \), we have

\[
\kappa \hat{P}(t_0) = A[R\hat{T}]P(t_0)
\]

where \( \kappa \) is a scale factor called the projective depth, and the camera position and orientation of the new
viewpoint is encoded in the 3 x 3 rotation matrix \( R \) and the translation \( T \) representing the rigid transformation that aligns the camera reference frames and \( A \) is the camera parameter matrix given by

\[
A = \begin{bmatrix}
\alpha & \gamma & u_0 \\
0 & \beta & v_0 \\
0 & 0 & 1
\end{bmatrix}
\]  

(2)

where \((u_0, v_0)\) are the coordinates of the principal point, \( \alpha = -f\kappa_u, \beta = -f\kappa_v \) are the focal lengths in the horizontal and vertical pixels, \( f \) is the focal length in mm, and \( \kappa_u, \kappa_v \) are the effective number of pixels per millimeter along the image axes, and \( \gamma \) is skewness of the image axes.

If we take the first viewpoint reference frame as the world reference frame, then the two views of the same action at corresponding time instants are related by

\[
\kappa'\hat{p}(t_0) = A[I \ 0] \hat{P}(t_0) \quad \text{and} \quad \kappa'\hat{p}(t_0) = A'[RT] \hat{P}(t_0) \quad \text{or}
\]

\[
\kappa'\hat{p}(t_0) = A'[RT] \hat{P}(t_0) = \kappa A' RA^{-1} \hat{p}(t_0) + A'T
\]

\[
\kappa'\hat{p}(t_0) = kH\hat{p}(t_0) + \epsilon' (3)
\]

where \( \epsilon' = AT \) and \( H = A' RA^{-1} \). Here we allow the internal parameters of the two cameras to be different. The above equation captures the constraint that the corresponding point \( \hat{p}(t_0) \) lies on the line going through the epipole \( AT \) and the point \( H\hat{p}(t_0) \). Expressing this collinearity in projective coordinates with the external product \( \hat{p}(t_0)(e' \Lambda H\hat{p}(t_0)) = 0 \), or

\[
\hat{p}^T(t_0)F\hat{p}(t_0) = 0
\]

(4)

where \( F \) is called the fundamental matrix. In the most general case, the only geometric information that can be computed from pairs of images is the fundamental matrix. Since equation 4 is homogeneous, the 3x3 matrix \( F \) can only be recovered up to a scale factor.

In fact, eight corresponding points in general positions are sufficient to solve \( F \) up to a unique scale factor as shown earliest by Longuet-Higgins [11] and popularized by Tsai and Huang[16]. Thus to recover the components of \( F \) we expand equation 4 and make \( f_{33} = 1 \) to form the linear system of equations

\[
x_j(t_0)x_i(t_0)f_{11} + x_j(t_0)y_i(t_0)f_{12} + x_j(t_0)f_{13} + y_i(t_0)x_i(t_0)f_{21} + y_i(t_0)y_i(t_0)f_{22} + y_i(t_0)f_{23} + x_i(t_0)f_{31} + y_i(t_0)f_{32} = -1
\]

(5)

Although several ways of computing the fundamental matrix have evolved in literature since[3, 12], the 8-point algorithm based on the above equations has been shown to be stable with simple normalization of coordinate points[9].

Traditionally, the recovery of \( F \) had led to the decomposition of \( F \) into rotation and translation components to recover motion and hence the structure of the 3d object. In the context of action recognition though, equation 4 states the following:

**Claim 1:** A set of eight corresponding points between corresponding cross-sections is sufficient to recover the viewpoint change between two action cylinders.

**Recognizing actions after recovering viewpoint**

Does the recovery of the viewpoint automatically enable the recognition of the action? Ordinarily, this would require the knowledge of the 3d shape of the object. Since no 3d shape information is known in both model and image action cylinders, the recovery of viewpoint alone is not sufficient to enable recognition. Notice, however, that Equation 4 is essentially a re-statement of the epipolar line constraint. That is, a point on the reference or model action cylinder that corresponds to a point on the given action cylinder is only known up to a line. If a given action cylinder is indeed depicting the same action from a different viewpoint and the chosen 8-point correspondence is correct, then we should expect a large number of features on the model action cylinder to lie on or be near some epipolar lines corresponding to features on the given action cylinder. That is, if we collect all the features from the model(M) and given action cylinders (N) at the corresponding time instants \( t_0 \) and \( t_0' \) into matrices, we have

\[
\hat{S}'(t_0) = \begin{bmatrix} x'_1(t_0) & x'_2(t_0) & \cdots & x'_N(t_0) \\ y'_1(t_0) & y'_2(t_0) & \cdots & y'_N(t_0) \end{bmatrix}
\]

\[
\hat{S}(t_0') = \begin{bmatrix} x_1(t_0) & x_2(t_0) & \cdots & x_M(t_0') \\ y_1(t_0) & y_2(t_0) & \cdots & y_M(t_0') \end{bmatrix}
\]

(6)

(7)

then the matrix product

\[
\hat{S}'(t_0)F\hat{S}(t_0')
\]

(8)

can be used to form a verification measure since it will have close-to-zero entries for all candidate corresponding points \((x_i(t_0), y_i(t_0))\) and \((x'_j(t_0), y'_j(t_0))\). Let \( n_{\text{zero}} \) be the number of such entries. Since a single point may match to more than one by this measure, a simple way to avoid this multiple count is to use the \( \min(n_{\text{zero}}, N, M) \) to obtain a normalized verification score as:

\[
\text{Verif}(t_0, t_0') = \frac{\min(n_{\text{zero}}, N, M)}{\min(N, M)}
\]

(9)
Thus given a pair of corresponding cross-sections on the two action cylinders, the viewpoint transformation can be recovered, and the cross-sections can be verified by projection using equation 9 above.

Since we focused on a single pair of cross-sections here, this can only recognize the shape (of the object undergoing motion) part of the action. If the time correspondence for all cross-sections \((t_i, t_i')\) was known, however, this process can be repeated to verify the entire action cylinder, one cross-section at a time. The final verification is then simply the sum of the pair-wise verifications, and the action is said to be recognized if the following normalized verification score exceeds a chosen threshold \(\tau\)

\[
\sum_i \text{Verif}(t_i, t_i') > \tau
\]

Alternately, when only the verified features of the model cylinder are retained and the action sequence re-synthesized, we should expect to see similarity with the original reference or model action sequence. In the results section, we show that this is indeed the case for correct recognition.

**Discussion**

The above result is significant since it implies that even without full 3D reconstruction, we can recognize the action through a simple verification constraint based on the epipolar line constraint. While accidental alignment of points is possible, such repeated alignment of projected points with present features throughout all cross-sections is unlikely for the incorrect pose. Also, unlike object recognition, where only the visible points in two views can be verified, points that disappear can re-appear at a later time and can still be recognized during the evolution of an action. Also, the approach taken here is more general than the one in [5] where the transformation recovered from two video sequences was a homography.

**Reducing search in correspondence through tracking**

Note that the corresponding points used in equation 5 need not come from a single pair of cross-sections. Equation 5 will hold for all corresponding points on corresponding cross-sections. This observation is significant because it implies that the search for 8 corresponding features can be reduced by tracking. In the extreme case, we can pick one starting feature pair from cross-sections \(S'(t_0)\) and \(S(t_0')\), and derive the remaining 7 features by noting the position of the starting feature in successive corresponding cross-sections through tracking. Several other ways of picking corresponding features are also possible, such as 4 in one cross-section, 3 in the other, all 8 in one cross-section, etc. In general, if the tracked features are well-separated, an accurate recovery of \(F\) can be ensured. Thus while tracking is not essential to our formulation for purposes of recognition, it can no doubt help in search reduction during recognition.

**Effect of occlusions and noise**

Occlusions are of two types in the case of action cylinder matching. First, the object undergoing motion itself may appear occluded (either due to actual occlusion or object detection errors). Secondly, a partial execution of an action can also be thought of as an occlusion. The former manifests as change in the shape of the individual cross-sections or equivalently, loss of features in these cross-sections, causing the individual verification score to be affected. The latter causes loss of cross-sections in the cylinder affecting the combined verification score. In general, since we initiate search for correspondence from the given action cylinder, but initiate verification from the model cylinder, tolerance to occlusions is achieved. Large occlusions, however, can still be a problem.

**3.2 Handling non-rigid and articulated object motion**

Non-rigid articulated object motions are characterized by separate frame-to-frame transformation. Since the viewpoint transformation is a function of the camera geometry and not the object motion, the above analysis carries through even when the object is non-rigid provided a pair of corresponding cross-sections can be found. Also, since the matching and verification of the cross-sections is based on the local shape at corresponding time instants, the non-rigidity of the object or its motion does not affect the verification process.

**4 The action recognition algorithm**

Putting the above results together, the overall action recognition algorithm consists of a pre-processing step followed by matching.

**Action cylinder generation and pre-processing**

To generate an action cylinder, the moving object is first extracted from the video. Since we assumed a single moving object, a simple background subtraction using frame difference with a background frame, was found sufficient. We have also experimented with other ways of segmenting objects including color-based segmentation, and simple frame-by-frame differencing. In general, we expect some segmentation errors so that a portion of the background may be included or a portion of the object may be missing in bit maps extracted. Some of the noisy portions of segmented objects can be merged through dilation. Since the verification is performed model to image cylinder-wise, the spurious
features do not affect as much in the final verification score. An edge detector is then applied over each frame image and the edges in the object-containing regions are retained. Curves are assembled from the edges, and corner features are extracted from curves as places where there is a sharp change in curvature. Each action is now represented by the curves and corners in each cross-section at successive time instants to serve as the action cylinder. We use the curves mainly to pick features for correspondence in an orderly manner, and also to statically display an action. This processing is repeated for all model actions and the resulting model cylinders stored in a library.

In addition to cylinder creation, we also extract salient features during the pre-processing stage to facilitate the search for corresponding features. The saliency was based on three factors: 1) the sharpness of included angle, 2) the length of the incoming lines, and 3) proximity to other features. The salient features are then tracked using a conventional optical flow algorithm. For each feature tracked, we note the cross-sections in which the tracked feature was also verified by an actually detected corner feature at the indicated location. This gave rise to a set of tracked salient features \( F_\ell = \{ F_{\ell}(t_j) \} \), where \( F_{\ell}(t_j) \) is the \( \ell \)th tracked feature in \( t_j \)th (time instant) cross-section. These features are stored as part of the model cylinder descriptions in the library.

**Matching algorithm:**

Given a new action, we generate its action cylinder and the set of tracked salient features as described above. We then match it against each model cylinder in the library using the following algorithm.

1. Pick a starting time instant \( t_0 \) on the given action cylinder. We usually choose a cross-section that has 8 or more salient features. To achieve tolerance to action segmentation errors though, different starting time instants may have to be tried.

2. Pick a corresponding time \( t'_0 \) on the model cylinder. For the first iteration \( t'_0 = 1 \).

3. Define correspondence for all cross-sections of the two cylinders. Since we are not considering velocity and acceleration changes here, we assume the time correspondence to be one-one up to a translation, i.e. we have \( t' - t'_0 = t - t_0 \).

4. Pick three salient features \( F'_1(t_0), F'_2(t_0), F'_3(t_0) \) in cross-section \( S'_1(t_0) \) on the given action cylinder. For each of these features, pick two other features at fixed (but possibly different) offsets in the tracked feature set. We retain 8 of these features as \( F'_1(t_1), F'_1(t_2), F'_1(t_3), F'_2(t_0), F'_2(t_2), F'_2(t_3), F'_3(t_0), F'_3(t_2) \).

5. Pick a set of 3 matching candidate features \( F_1(t'_0), F_2(t'_0), F_3(t'_0) \) from cross-section \( S(t'_0) \) on the model cylinder. Pick the remaining 5 features from their tracked versions at corresponding times as \( F_1(t'_2), F_1(t'_3), F_2(t'_4), F_3(t'_2), F_3(t'_3) \). Since we are not considering velocity and acceleration changes here, we have \( t'_3 - t'_2 = t_4 - t_3 \).

6. Use the corresponding features to solve for the fundamental matrix using Equation 5.

7. Verify the correctness of the solution using equation 10. If the verification score is above the acceptable threshold, declare action recognized. If not, repeat steps 4 through 6 until all valid feature pairings are tried.

8. If these fail, repeat steps 2 through 7 with another corresponding time instant on the model cylinder. Declare mis-recognition if all else fails.

**Complexity Analysis**

To analyze the complexity of the above action recognition algorithm, let \( m_i, i = 1, M \) be the number of features in cross-section \( i \) in the model cylinder (of \( M \) cross-sections). Let \( n_j, j = 1, N \) be the number of features in cross-section \( j \) in the image cylinder. The search for matching features within a pair of cross-sections takes \( O(m_i^2 n_j^2) \), since the remaining 5 features are chosen from fixed cross-sections in the tracked salient features. The overall search, therefore, is \( O(n^8) \) for a fixed choice of the starting cross-section \( t_0 \) in the given action cylinder.

**5 Results**

To test the recognition of actions, we experimented with several 3d objects, ranging from synthetic rigid objects under known viewpoints, to articulated objects (hands) executing dance-like motions to completely non-rigid such as flower stalks and clothes moving in the wind. There are currently over 60 objects in our collection. Each object was made to exhibit between 5-20 actions. For each action type, different instances were generated by observing the action at different times, and under different viewpoints and at different times for a total of 600 actions in the database. Since object and action segmentation were not the focus of this paper, we made the experimental set up in most cases have a white or dark background. The cameras were always fixed, and the object was made to undergo the action. In all these experiments, the illumination set up for the different instances was more or less the same. However, illumination variations could still be seen in the different videos of the same action due to camera and viewpoint differences.

**Examples**

We now illustrate results of action recognition through a few examples. Some of these were based on an earlier implementation of the matching algorithm in which we experimented with several valid feature
segmentation, since an action has a large number of features spanning multiple time instants, it was always to object segmentation errors, these cylinders have a portion of the background mixed. Even with imperfect feature detection, and loss of features due to segmentation, since an action has a large number of features spanning multiple time instants, it was always possible to find features for correspondence as well as pose verification. For this example, we used an earlier version of the algorithm that chose the set of eight corresponding points on the model and image action cylinders from a single cross-section at time instant 5. The positions of these points that gave rise to a valid fundamental matrix that was verified by the model action cylinder features are depicted by circles in Figure 2c and d. For the fundamental matrix computed through this correspondence, nearly 12,000 of the 15,000 features from 60 cross-sections of the action sequence (the first 2 seconds of the sequence were used) were verified as seen from Figure 2e and f. Figure 2e shows the original model features on the model action cylinder, while Figure 2f shows the verified model features retained. By animating the verified features in their respective frames, we were able to reconstruct an action sequence that resembled the model action sequence.

Table 1: Illustration of effectiveness of action recognition in the action database. It indicates the number of similar actions to each of the model actions under changes in viewpoint.

<table>
<thead>
<tr>
<th>Object</th>
<th>Action type</th>
<th># viewpoint difference</th>
<th># correct</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>toy</td>
<td>rigid</td>
<td>8</td>
<td>8</td>
<td>100%</td>
</tr>
<tr>
<td>hand</td>
<td>articulated</td>
<td>7</td>
<td>6</td>
<td>85%</td>
</tr>
<tr>
<td>flower</td>
<td>non-rigid</td>
<td>5</td>
<td>3</td>
<td>60%</td>
</tr>
<tr>
<td>pencil</td>
<td>rigid</td>
<td>13</td>
<td>11</td>
<td>84.6%</td>
</tr>
<tr>
<td>hand</td>
<td>articulated</td>
<td>4</td>
<td>4</td>
<td>100%</td>
</tr>
<tr>
<td>hand</td>
<td>rigid</td>
<td>7</td>
<td>6</td>
<td>85%</td>
</tr>
<tr>
<td>bottle</td>
<td>rigid</td>
<td>5</td>
<td>4</td>
<td>80%</td>
</tr>
<tr>
<td>toy</td>
<td>non-rigid</td>
<td>10</td>
<td>9</td>
<td>90%</td>
</tr>
</tbody>
</table>

Errors in recognition

In the next set of experiments, we tested the accuracy of action recognition against the database of model actions. Specifically, for each of the 600 actions in the database, we noted the number of actions that were perceptually similar to the given action under changes in viewpoint. We then used each of these actions as a test action, and noted the matching actions returned in the top n. The value of n varied based on the action. That is, for actions which were known to have n similar actions in the database, the top n matches were retained. If all the n matches returned were correct, then this was considered 100% recall. The number of mismatches in the top n matches was taken as an indication of precision. The results are indicated in Table 1 for some representative objects and their actions tested. As can be seen, the action recognition algorithm possesses good recall for the actions tested.

Errors in recognition

The recognition errors could be accounted for by several factors: (a) tracking errors, (b) time correspondence errors, (c) lack of common corresponding points under changes in viewpoint, and finally, (d) object segmentation errors. Since we have used a sparse feature-based representation of the action, tracking can sometimes cause problems where missing features occur in successive frames, causing problems in the generation of corresponding points. In such cases, however, we predict the positions of features using the motion computed in previous frames. Next, the lack of common corresponding points between the two is an obvious limitation. The prediction in tracking can sometimes account for these missing or occluded features. Fortunately, there are plenty of cross-sections to try for an action sequence. Finally, object segmentation errors can cause a wrong action to be verified when the verification picks up more features from the background rather than the object region in the epipolar lines-based test. Often though, we have observed the spurious features to affect the verification score without changing the result on the best matching action. A case where this hap-
pens is shown in Figure 3, where the background features are also matched in the verification step. Better object segmentation methods such as color-based object selection can alleviate this problem.

**Conclusions**

In this paper we have analyzed the effect of viewpoint changes on action representation and recognition. The action cylinder representation was shown to be computationally simple as well as versatile in handling actions of rigid and non-rigid objects.

**References**


Figure 2: Illustration of action recognition under changes in viewpoint. — Rigid object case. (a) given action cylinder of a rotating toy. (b) model cylinder of the rotating toy seen from a different viewpoint. (c)-(d) Matching salient features on the two cylinders. (e) Features on the model action cylinder projected into image plane. (f) Features on the model action cylinder that found a match to features on the given cylinder in successful verification using the corresponding features of (c)-(d).

Figure 3: Illustration of action recognition under changes in viewpoint. — Case of articulated/non-rigid motion. (a) given action cylinder of a dancing hand. (b) model cylinder of the dancing action seen from a different viewpoint. (c)-(d) Tracked salient features from two cross-sections on the given cylinder. (e)-(f) tracked salient features on the model cylinder that found a match. (g) features on the model action cylinder projected into image plane. (h) Features on the model action cylinder that found a match to features on the given cylinder in successful verification using the corresponding features of (c)-(f).