Detection of Manipulation Action Consequences (MAC)

Yezhou Yang yzyang@cs.umd.edu
Cornelia Fermüller fer@umiacs.umd.edu
Yiannis Aloimonos yiannis@cs.umd.edu
Computer Vision Lab, University of Maryland, College Park, MD 20742, USA

Abstract

The problem of action recognition and human activity has been an active research area in Computer Vision and Robotics. While full-body motions can be characterized by movement and change of posture, no characterization, that holds invariance, has yet been proposed for the description of manipulation actions. We propose that a fundamental concept in understanding such actions, are the ***consequences of actions****. There is a small set of fundamental primitive action consequences that provides a systematic high-level classification of manipulation actions. In this paper a technique is developed to recognize these action consequences. At the heart of the technique lies a novel active tracking and segmentation method that monitors the changes in appearance and topological structure of the manipulated object. These are then used in a visual semantic graph (VSG) based procedure applied to the time sequence of the monitored object to recognize the action consequence. We provide a new dataset, called* ***M****anipulation* ***A****ction* ***C****onsequences (MAC 1.0), which can serve as testbed for other studies on this topic. Several experiments on this dataset demonstrates that our method can robustly track objects and detect their deformations and division during the manipulation. Quantitative tests prove the effectiveness and efficiency of the method.*

1. Introduction

Visual recognition is the process through which intelligent agents associate a visual observation to a concept from their memory. In most cases, the concept either corresponds to a term in natural language, or an explicit definition in natural language. Most research in Computer Vision has focused on two concepts: objects and actions; humans, faces and scenes can be regarded as special cases of objects. Object and action recognition are indeed crucial since they are the fundamental building blocks for an intelligent agent to semantically understand its observations.

When it comes to understanding actions of manipulation, the movement of the body (especially the hands) is not a very good characteristic feature. There is great variability in the way humans carry out such actions. It has been realized that such actions are better described by involving a number of quantities. Besides the motion trajectories, the objects involved, the hand pose, and the spatial relations between the body and the objects under influence, provide information about the action. In this work we want to bring attention to another concept, the action consequence. It describes the transformation of the object during the manipulation. For example during a CUT or a SPLIT action an object is divided into segments, during a GLUE or a MERGE action two objects are combined into one, etc.

The recognition and understanding of human manipulation actions recently has attracted the attention of Computer Vision and Robotics researchers because of their critical role in human behavior analysis. Moreover, they naturally relate to both, the movement involved in the action and the objects. However, so far researchers have not considered that the most crucial cue in describing manipulation actions is actually not the movement nor the specific object under influence, but the object centric action consequence. We can come up with examples, where two actions involve the same tool and same object under influence, and the motions of the hands are similar, for example in "cutting a piece of meat" vs. "poking a hole into the meat". Their consequences are different. In such cases, the action consequence is the key in differentiating the actions. Thus, to fully understand manipulation actions, the intelligent system should be able to determine the object centric consequences.

Few researchers have addressed the problem of action consequences due to the difficulties involved. The main challenge comes from the monitoring process, which calls for the ability to continuously check the topological and appearance changes of the object-under-manipulation. Previous studies of visual tracking have considered challenging situations, such as non-rigid objects [8], adaptive appearance model [12], and tracking of multiple objects with occlusions [24], but none can deal with the difficulties involved in detecting the possible changes on objects during manipulation. In this paper, for the first time, a system

is implemented to conquer these difficulties and eventually achieve robust action consequence detection.

2. Why Consequences and Fundamental Types

Recognizing human actions has been an active research area in Computer Vision [10]. Several excellent surveys on the topic of visual recognition are available ([21], [29]). Most work on visual action analysis has been devoted to the study of movement and change of posture, such as walking, running etc. The dominant approaches to the recognition of single actions compute as descriptors statistics of spatio-temporal interest points ([16], [31]) and flow in video volumes, or represent short actions by stacks of silhouettes ([4], [34]). Approaches to more complex, longer actions employ parametric approaches, such as Hidden Markov Models [13], Linear Dynamical Systems [26] or Non-linear Dynamical Systems [7], which are defined on extracted features. There are a few recent studies on human manipulation actions ([30], [14], [27]), but they do not consider action consequences for the interpretation of manipulation actions. Works like [33] emphasize the role of object perception in action or pose recognition, but they focus on object labels, not object-centric consequences.

How do humans understand, recognize, and even replicate manipulation actions? Psychological studies on human manners ([18] etc.) have pointed out the importance of manipulation action consequences for both understanding human cognition and intelligent system research. **Actions, by their very nature, are goal oriented.** When we perform an action, we always have a goal in mind, and the goal affects the action. Similarly, when we try to recognize an action, we also keep a goal in mind. The close relation between the movement during the action and goal is reflected also in language. For example, the word "CUT" denotes both the action in which hands move up and down or in and out with sharp bladed tools, and the consequence of the action, namely that the object is separated. Very often, we can recognize an action purely by the goal satisfaction, and even neglect the motion or the tools used. For example, we may observe a human carry out movement with a knife, that is "up and down", but if the object remains as one whole, we won't draw the conclusion that a "CUT" action has been performed. Only when the goal of the recognition process, here "DIVIDE", is detected, the goal satisfaction is reached and a "CUT" action is confirmed. An intelligent system should have the ability to detect the consequences of manipulation actions, in order to check the goal of actions.

In addition, experiments conducted in neuronscience [25] show that a monkey's mirror neuron system fires when a hand/object interaction is observed, and it will not fire when a similar movement is observed without hand/object interaction. Recent experiments [9] further showed that the mirror neuron regions responding to the sight of actions responded more during the observation of goal-directed actions than similar movements not directed at goals. These evidences support the idea of goal matching, as well as the crucial role of action consequence in the understanding of manipulation actions.

Taking an object-centric point of view, manipulation actions can be classified into six categories according how the object is transformed during the manipulation, or in other words what consequence the action has on the object. These categories are: DIVIDE, ASSEMBLE, CREATE, CONSUME, TRANSFER, and DEFORM. Each of these categories is denoted by a term that has a clear semantic meaning in natural language given as follows:

- DIVIDE: one object breaks into two objects, or two attached objects break the attachment;
- ASSEMBLE: two objects merge into one object, or two objects build an attachment between them;
- CREATE: an object is brought to, or emerges in the visual space;
- CONSUME: an object disappears from the visual space;
- TRANSFER: an object is moved from one location to another location;
- DEFORM: an object has an appearance change.

To describe these action categories we need a formalism. We use the visual semantic graph (VSG) inspired from the work of Aksoy et. al [1]. This formalism takes as input computed object segments, their spatial relationship, and temporal relationship over consecutive frames. To provide the symbols for the VSG, an active monitoring process (discussed in sec. 4) is required for the purpose of (1) tracking the object to obtain temporal correspondence, and (2) segmenting the object to obtain its topological structure and appearance model. This active monitoring (consisting of segmentation and tracking) is related to studies on active segmentation [20], and stochastic tracking ([11] etc.).

3. Visual Semantic Graph (VSG)

To define object-centric action consequences, a graph representation is used. Every frame in the video is described by a Visual Semantic Graph (VSG), which is represented by an undirected graph $G(V, E, P)$. The vertex set $|V|$ represents the set of semantically meaningful segments, the edge set $|E|$ represents the spatial relations between any of the two segments. Two segments are connected when they share parts of their borders, or when one of the segments is contained in the other. If two nodes $v_1, v_2 \in V$ are connected, $E(v_1, v_2) = 1$, otherwise, $E(v_1, v_2) = 0$. In addition, every node $v \in V$ is associated with a set of properties $P(v)$, that describes the attributes of the segment. This set of properties provides additional information to discriminate the different categories, and in principle many properties are possible. Here we use location, shape, and color.

We need to compute the changes of the object over time.

In our formulation this is expressed as the change in the VSG. At any time instance t, we consider two consecutive VSGs, the VSG at time $t-1$, denoted as $G_a(V_a, E_a, P_a)$ and the VSG at time t, denoted as $G_z(V_z, E_z, P_z)$. We then define the following four consequences, where $\rightarrow$ is used to denote the temporal correspondence between two vertices, $\nrightarrow$ is used to denote no correspondence:

- DIVIDE: $\{\exists v_1 \in V_a; v_2, v_3 \in V_z | v_1 \rightarrow v_2, v_1 \rightarrow v_3)\}$ *or* $\{\exists v_1, v_2 \in V_a; v_3, v_4 \in V_z | E_a(v_1, v_2) = 1, E_z(v_3, v_4) = 0, v_1 \rightarrow v_3, v_2 \rightarrow v_4\}$ **Condition (1)**
- ASSEMBLE: $\{\exists v_1, v_2 \in V_a; v_3 \in V_z | v_1 \rightarrow v_3, v_2 \rightarrow v_3\}$ *or* $\{\exists v_1, v_2 \in V_a; v_3, v_4 \in V_z | E_a(v_1, v_2) = 0, E_z(v_3, v_4) = 1, v_1 \rightarrow v_3, v_2 \rightarrow v_4\}$ **Condition (2)**
- CREATE:$\{\forall v \in V_a; \exists v_1 \in V_z | v \nrightarrow v_1\}$ **Condition (3)**
- CONSUME:$\{\forall v \in V_z; \exists v_1 \in V_a | v \nrightarrow v_1\}$ **Condition(4)**

While the above actions can be defined purely on the basis of topological changes, there are no such changes for TRANSFER and DEFORM. Therefore, we have to define them through changes in property. In the following definitions, P^L represents properties of location, and P^S represents properties of appearance (shape, color, etc.).

- TRANSFER:$\{\exists v_1 \in V_a; v_2 \in V_z | P_a^L(v_1) \neq P_z^L(v_2)\}$ **Condition (5)**
- DEFORM: $\{\exists v_1 \in V_a; v_2 \in V_z | P_a^S(v_1) \neq P_z^S(v_2)\}$ **Condition (6)**

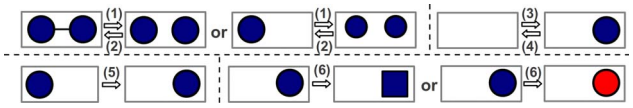

Figure 1: Graphical illustration of the changes for **Condition (1-6)**.

A graphical illustration for **Condition (1-6)** is shown in Fig. 1. Sec. 4 describes how we find the primitives used in the graph. A new active segmentation and tracking method is introduced to 1) find correspondences ($\rightarrow$) between V_a and V_z; 2) monitor location property P^L and appearance property P^S in the VSG.

The procedure for computing action consequences, first decides on whether there is a topological change between G_a and G_z. If yes, the system checks whether **Condition (1)** to **Condition (4)** are fulfilled and returns the corresponding consequence. If no, the system then checks whether **Condition (5)** or **Condition (6)** is fulfilled. If both of them are not met, no consequence is detected.

4. Active Segmentation and Tracking

Previously, researchers have treated segmentation and tracking as two different problems. Here we propose a new method combining the two tasks to obtain the information necessary to monitor the objects under influence. Our methods combines stochastic tracking [11] with a fixation based active segmentation [20]. The tracking module provides a number of tracked points. The locations of these points are used to define an area of interest and a fixation point for the segmentation, and the color in their immediate surroundings are used in the data term of the segmentation module. The segmentation module segments the object, and based on the segmentation, updates the appearance model for the tracker. Fig 2 illustrates the method over time, which is a dynamically closed-loop process. We next describe the attention based segmentation (sec. 4.1 - 4.4), and then the segmentation guided tracking (sec. 4.5).

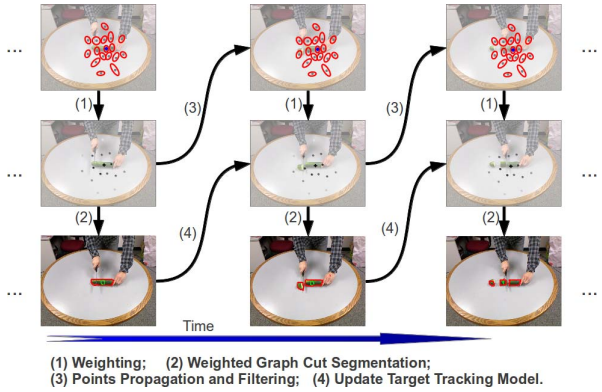

Figure 2: Flow chart of the proposed active segmentation and tracking method for object monitoring.

The proposed method meets two challenging requirements, necessary to detect action consequences: 1) the system is able to track and segment objects when the shape or color (appearance) changes; 2) the system is also able to track and segment objects when they are divided into pieces. Experiments in sec. 5.1 show that our method can handle these requirements, while systems implementing independently tracking and segmentation cannot.

4.1. The Attention Field

The idea underlying our approach is, that first a process of visual attention selects an area of interest. Segmentation then is considered the process that separates the area selected by visual attention from background by finding closed contours that best separate the regions. The minimization uses a color model for the data term and edges in the regularization term. To achieve a minimization that is very robust to the length of the boundary, edges are weighted with their distance from the fixation center.

Visual attention, the process of driving an agent's attention to a certain area, is based on both bottom-up processes defined on low level visual features, and top-down processes influenced by the agent's previous experience [28]. Inspired by the work of Yang et al. [32], instead of using a single fixation point in the active segmentation [20], here we use a weighted sample set $S = \{(s^{(n)}, \pi^{(n)}) | n = 1 \dots N\}$

to represent the attention field around the fixation point ($N = 500$ in practice). Each sample consists of an element s from the set of tracked points and a corresponding discrete weight π where $\sum_{n=1}^{N} \pi^{(n)} = 1$.

Generally, any appearance model can be used to represent the local visual information around each point. We choose to use a color histogram with a dynamic sampling area defined by an ellipse. To compute the color distribution, every point is represented by an ellipse, $s = \{x, y, \dot{x}, \dot{y}, H_x, H_y, \dot{H}_x, \dot{H}_y, \}$ where x and y denote the location, $\dot{x}$ and $\dot{y}$ the motion, H_x, H_y the length of the half axes, and $\dot{H}_x$, $\dot{H}_y$ the changes in the axes.

4.2. Color Distribution Model

To make the color model invariant to various textures or patterns, a color distribution model is used. A function $h(x_i)$ is defined to create a color histogram, which assigns one of the m-bins to a giving color at location x_i. To make the algorithm less sensitive to lighting conditions, the HSV color space is used with less sensitivity in the V channel ($8 \times 8 \times 4$ bins). The color distribution for each fixation point $s^{(n)}$ is computed as:

$$p(s^{(n)})^{(u)} = \gamma \sum_{i=1}^{I} k(||y - x_i||)\delta[h(x_i) - u], \qquad (1)$$

where $u = 1 \ldots m$, $\delta(.)$ is the Kronecker delta function, and γ is the normalization term $\gamma = \frac{1}{\sum_{i=1}^{I} k(||y-x_i||)}$. $k(.)$ is a weighting function designed from the intuition that not all pixels in the sampling region are equally important for describing the color model. Specifically, pixels that are farther away from the point are assigned smaller weights, $k(r) = \begin{cases} 1 - r^2 & \text{if } r < a \\ 0 & \text{otherwise} \end{cases}$, where the parameter a is used to adapt the size of the region, and r is the distance from the fixation point. By applying the weighting function, we increase the robustness of the color distribution by weakening the influence from boundary pixels, which possibly belong to the background or are occluded.

4.3. Weights of the Tracked Point Set

In the following weighted graph cut approach, every sample is weighted by comparing its color distribution with the one of the fixation point. Initially a fixation point is selected, later the fixation point is computed as the center of the tracked point set. Let's call the distribution at the fixation point q, and the histogram of the n^{th} tracked point, $p(s^{(n)})$. In assigning weights $\pi^{(n)}$ to the tracked points we want to favor points whose color distribution is similar to the fixation point. We use the Bhattacharyya coefficient $\rho[p, q] = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}}$ with m the number of bins to weigh points by a Gaussian with variance σ ($\sigma = 0.2$ in practice) and define $\pi^{(n)}$ as:

$$\pi^{(n)} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1-\rho[p(s^{(n)}),q]}{2\sigma^2}}. \qquad (2)$$

4.4. Weighted Graph Cut

The segmentation is formulated as a minimization that is solved using graph cut. The unary terms are defined on the tracked points on the basis of their color, and the binary terms are defined on all points on the basis of edge information. To obtain the edge information, in each frame, we compute a probabilistic edge map I_E using the Canny edge detector. Consider every pixel $x \in X$ in this edge map as a node in a graph. Denoting the set of all the edges connecting neighboring nodes in the graph as Ω, and using the label set $l = 0, 1$ to indicate whether a pixel x is "inside" ($l_x = 0$) or "outside" ($l_x = 1$), we need to find a labeling $f(X) \longmapsto l$, that minimizes the energy function:

$$Q(f) = \sum_{x \in X} U_x(l_x) + \lambda \sum_{(x,y) \in \Omega} V_{x,y}\delta(l_x, l_y). \qquad (3)$$

$V_{x,y}$ is the cost of assigning different labels to neighboring pixels x and y, which we defines as $V_{x,y} = e^{-\eta I_{E,xy}} + k$, with $\delta(l_x, l_y) = \begin{cases} 1 & \text{if } l_x \neq l_y \\ 0 & \text{otherwise} \end{cases}$, $\lambda = 1, \eta = 1000, k = 10^{-16}$, $I_{E,xy} = (I_E(x)/R_x + I_E(y)/R_y)/2$, R_x, R_y are the euclidean distances between the x, y and the center of the tracked point set S_t. We use them as weights to make the segmentation robust to the length of the contours.

$U_x(l_x)$ is the cost of assigning label l_x to pixel x. In our system, we have a set of points S_t, and for each sample $s^{(n)}$, there is a weight $\pi^{(n)}$. The weight itself indicates the likelihood that the area around that fixation point belongs to the "inside" of the object. It becomes straightforward to assign weights $\pi^{(n)}$ to the pixel $s^{(n)}$, which are tracked points as follows: $U_x(l_x) = \begin{cases} N\pi^{(n)} & \text{if } l_x = 1 \\ 0 & \text{otherwise} \end{cases}$. We assume that pixels on the boundary of a frame are "outside" of the object, and assign to them a large weight $W = 10^{10}$: $U_x(l_x) = \begin{cases} W & \text{if } l_x = 0 \\ 0 & \text{otherwise} \end{cases}$. Using this formulation, we run a graph cut algorithm [5] on each frame. Fig. 3a illustrates the procedure on a texture-rich natural image from the Berkeley segmentation dataset [19].

Two critical limitations of the previous active segmentation method [20] in practice are: 1) the segmentation performance largely varies under different initial fixation points; 2) the segmentation performance also is strongly affected by texture edges, which often leads to a segmentation of object parts. Fig. 3b shows that our proposed segmentation method is robust to the choice of initial fixation point, and only weakly affected by texture edges.

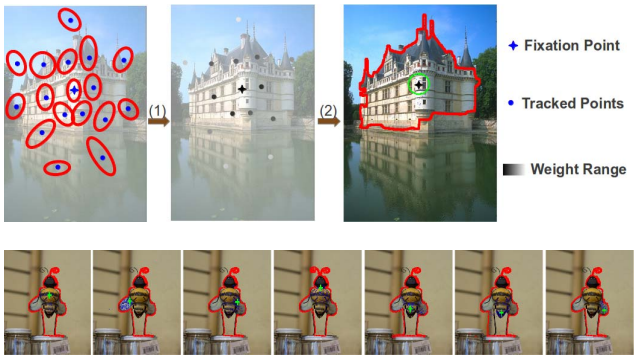

Figure 3: **Upper**: (1) Sampling of tracked points sampling and filtering; (2) Weighted graph cut. **Lower**: Segmentation with different initial fixations. Green Cross: initial fixation.

4.5. Active Tracking

At the very beginning of the monitoring process, a Gaussian sampling with mean at the initial fixation point and variances σ_x, σ_y is used to generate the initial point set S_0. When a new frame comes in, the point set is propagated through a stochastic tracking paradigm:

$$s_t = As_{t-1} + w_{t-1}, \tag{4}$$

where A denotes the deterministic, and w_{t-1} the stochastic component. In our implementation, we have considered a first order model for A, which assumes that the object is moving with constant velocity. The reader is referred to [23] for details. The complete algorithm is given in Algorithm 1

Algorithm 1 Active tracking and segmentation

Require: Given the tracked point set S_{t-1} and the target model q_{t-1}, perform the following steps:

1. **SELECT** N samples from the set S_{t-1} with probability $\pi_{t-1}^{(n)}$. Fixation points with a high weight may be chosen several times, leading to identical copies, while others with relatively low weights may not be chosen at all. Denote the resulting set as S'_{t-1};
2. **PROPAGATE** each sample from S'_{t-1} by a linear stochastic differential eq. 4. Denote the new set as S_t
3. **OBSERVE** the color distributions for each sample of S_t using eq. 1. Weigh each sample using eq. 2.
4. **SEGMENTATION** using the weighted sample set. Apply the weighted graph cut algorithm described in sec. 4.4. and get the segmented object area M.
5. **UPDATE** the target distribution q_{t-1} by the area M to achieve the new target distribution q_t.

4.6. Incorporating Depth and Optical Flow

It is easy to extend our algorithm to incorporate depth (for example from Kinect) or image motion flow information. Depth information can be used in a straightforward way during two crucial steps. 1) As described in sec. 4.2, we can add in depth information as another channel in the distribution model. In preliminary experiments we used 8 bins for the depth, to obtain in RGBD space a model with $8 \times 8 \times 4 \times 8$ bins. 2) Depth can be used to achieve cleaner edge maps, I_E, in the segmentation step 4.4.

Optical flow can be incorporated to provide cues for the system to predict the movement of edges to be used for the segmentation step in the next iteration, and the movement of the points in the tracking step. We performed some experiments using the optical flow estimation method proposed by Brox [6] and the improved implementation by Liu [17].

Optical flow was used in the segmentation by first predicting the contour of the object in the next frame, and then fusing it with the next frame's edge map. Fig. 4a shows an example of an edge map improved by optical flow. Optical flow was incorporated into tracking by replacing the first order velocity components for each tracked point in matrix A (eq. 4) by its flow component. Fig. 4b shows that the optical flow drives the tracked points to move along the flow vectors into the next frame.

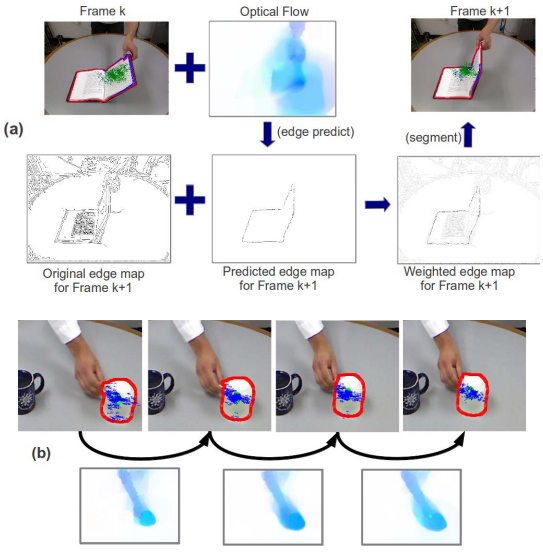

Figure 4: (a): Incorporating optical flow into segmentation. (b): Incorporating optical flow into tracking.

5. Experiments

5.1. Deformation and Division

To show that our method can segment challenging cases, we first demonstrate its performance for the case of de-

forming and dividing objects. Fig. 5a shows results for a sequence with the main object deforming, and Fig. 5b for a synthetic sequence with the main object dividing. The ability to handle deformations comes from the updating of the target model using the segmentation of previous frames. The ability to handle division comes from the tracked point set that is used to represent the attention field (sec. 4.1), which guides the weighted graph cut algorithm (sec. 4.4).

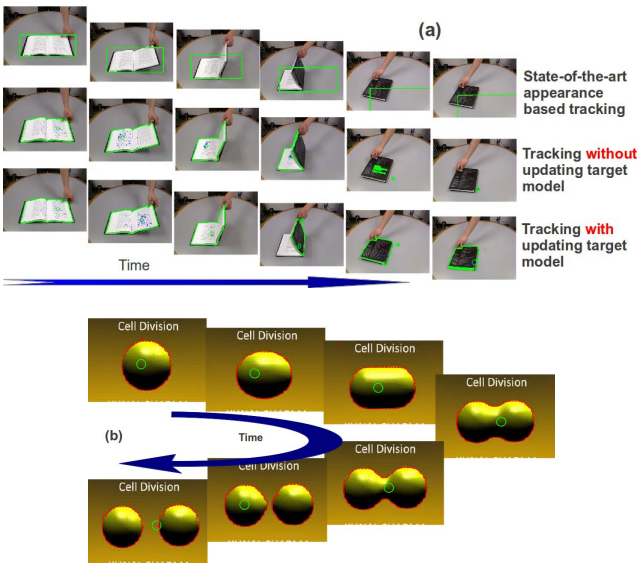

Figure 5: (a): Deformation Invariance: upper: state-of-the-art appearance based tracking [2]; middle: tracking without updating target model; lower: updating target model. (b): Division Invariance: synthetic cell division sequence.

5.2. The MAC 1.0 Dataset

To quantitatively test our method, we collected a dataset of several RGB+Depth image sequences of humans performing different manipulation actions. In addition, several sequences from other publicly available datasets ([1], [22] and [15]) were included to increase the variability and make it more challenging. Since the two action consequences CREATE and CONSUME (sec.2) relate to the existence of the object and would require a higher level attention mechanism, which is out of this paper's scope, we did not consider them. For the other four consequences, six sequences were collected each to make the first Manipulation Action Consequence (MAC 1.0) dataset.[1].

5.3. Consequence Detection on MAC 1.0

We first evaluated the method's ability in detecting the various consequences. Consequences happen in an event based manner. In our description, a consequence is detected using the VSG graph at a point in time, if between two consecutive image frames one of the conditions listed in sec. 3 is met. For example, a consequence is detected for the case of DIVIDE, when one segment becomes two segments in the next frame (Fig. 6), or for the case of DEFORM, when one appearance model changes to another (Fig. 8). We obtained ground truth by asking people not familiar with the purpose to label the sequences in MAC 1.0.

Fig. 6, 7, 8 show typical example active segmentation and tracking, the VSG graph, and the corresponding measures used to identify the different action consequences, as well as the final detection result along the time-line are illustrated. Specifically, for DIVIDE we monitor the change in the number of segments, for ASSEMBLE we monitor the minimum Euclidean distance between the contours of segments, for DEFORM we monitor the change of appearance (color histogram and shape context [3]) of the segment, and for TRANSFER we monitor the velocity of the object. Each of the measurements is normalized to the range of $[0, 1]$ for the ROC analysis. The detection is evaluated, by counting the correct detections over the sequence. For example, for the case of DIVIDE, at any point in time we have either the detection, "not divided" or "divided". For the case of ASSEMBLE , we have either the detection "two parts assembled" or "nothing assembled", and for DEFORM, we have either "deformed" or "nor deformed". The ROC curves obtained are shown in Fig. 9. The graphs indicate that our method is able to correctly detect most of the consequences. Several failures point out the limitations of our method as well. For example, for the PAPER-JHU sequence the method has errors in detecting DIVIDE, because the part that was cut out, connects visually with the rest of the paper. For the CLOSE-BOTTLE sequence our method fails for ASSEMBLE because the small bottle cap is occluded by the hand. However, our method detects that an ASSEMBLE event happened after the hand move away.

5.4. Video Classification on MAC 1.0

We also evaluated our method on the problem of classification, although the problem of consequence detection is quite different from the problem of video classification. We compared our method with the state-of-the-art STIP + Bag of Words + classification (SVM or Naive Bayes). The STIP features for each video in the MAC 1.0 dataset were computed using the method described in [16]. For classification we used a bag of words + SVM and a Naive Bayes method. The dictionary codebook size was 1000, and a polynomial kernel SVM with a leave-one-out cross validation setting was used. Fig. 10 shows that our method dramatically outperforms the typical STIP + Bag of Words + SVM and the Naive Bayes learning methods. However, this does not come as a surprise. The different video sequences in an action consequence class contain different objects and

[1]The dataset is available at www.umiacs.umd.edu/~yzyang.

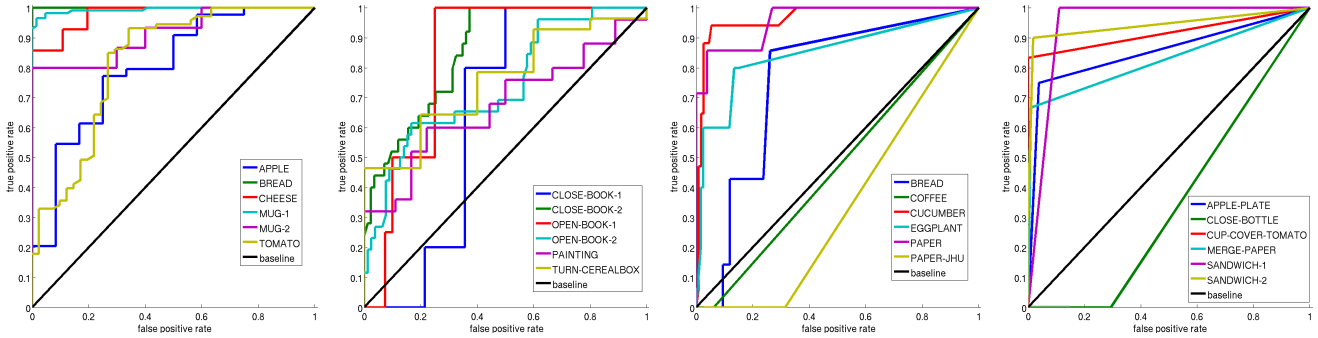

Figure 9: ROC curve of each sequence by categories: (a) TRANSFER, (b) DEFORM, (c) DIVIDE, and (d) ASSEMBLE.

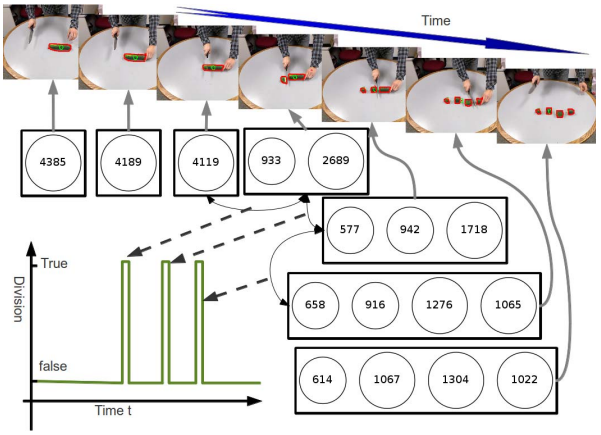

Figure 6: "Division" detection on "cut cucumber" sequence. Upper row: Original sequence with segmentation and tracking; Middle and lower right: VSG representations; Lower left: Division consequences detection.

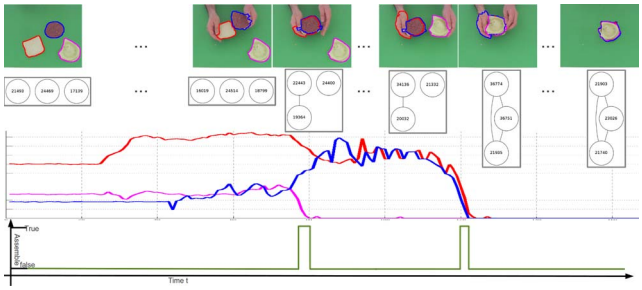

Figure 7: "Assemble" detection on "make sandwich 1" sequence; 1st row: Original sequence with segmentation and tracking; 2nd row: VSG representation; 3rd row: Distance between each two segments (red line: bread and cheese, magenta line: bread and meat, blue line: cheese and meat; 4th row: Assemble consequence detection.

different actions and thus different visual features, and are

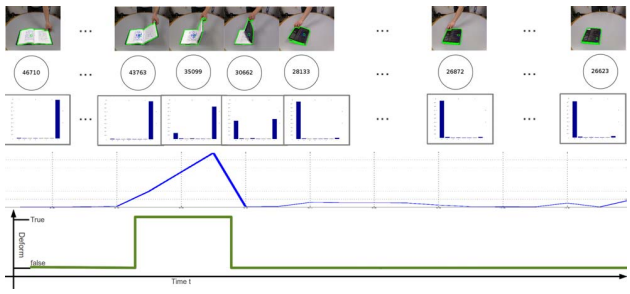

Figure 8: "Deformation" detection on "close book 1" sequence; 1st row: Original sequence with segmentation and tracking; 2nd row: VSG representation; 3rd row: appearance description (here color histogram) of each segment; 4th row: measurement of appearance change; 5th row: Deformation consequence detection.

therefore not well suited for standard classification. On the other hand, our method has been specifically designed for the detection of manipulation action consequences all the way from low-level signal processing through the mid-level semantic representation to high-level reasoning. Moreover, different from a learning based method, it does not rely on training data. After all, the method stems from the insight of manipulation action consequences.

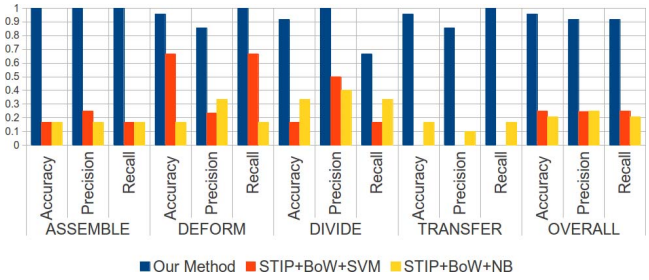

Figure 10: Video classification performance comparison.

6. Discussion and Future Works

A system for detecting action consequences and classifying videos of manipulation action according to action consequences has been proposed. A dataset has been provided, which includes both data that we collected and eligible manipulation action video sequences from other publicly available datasets. Experimental results were performed that validate our method, and at the same time point out several weaknesses for future improvement.

For example, to avoid the influence from the manipulating hands, especially occlusions caused by hands, a hand detection and segmentation algorithm can be applied. Then we can design a hallucination process to complete the contour of the occluded object under manipulation. Preliminary results are shown in Fig. 11. However, resolving the ambiguity between occlusion and deformation from visual analysis is a difficult task that requires further attention.

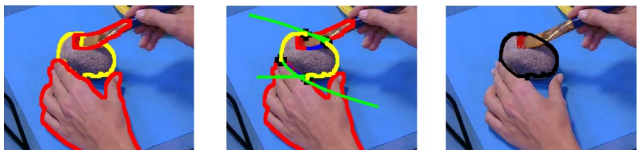

Figure 11: A hallucination process of contour completion (paint stone sequence in MAC 1.0). Left: original segments; Middle: contour hallucination with second order polynomials fitting (green lines); Right: final hallucinated contour.

7. Acknowledgements

The support of the European Union under the Cognitive Systems program (project POETICON++) and the National Science Foundation under the Cyberphysical Systems Program is gratefully acknowledged. Yezhou Yang has been supported in part by the Qualcomm Innovation Fellowship.

References

[1] E. Aksoy, A. Abramov, J. Dörr, K. Ning, B. Dellen, and F. Wörgötter. Learning the semantics of object–action relations by observation. *The International Journal of Robotics Research*, 30(10):1229–1249, 2011. 2, 6

[2] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1830–1837. IEEE, 2012. 6

[3] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. *Advances in neural information processing systems*, pages 831–837, 2001. 6

[4] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, volume 2, pages 1395–1402, 2005. 2

[5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI, IEEE Transactions on*, 26(9):1124–1137, 2004. 4

[6] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *ECCV*, pages 25–36, 2004. 5

[7] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *CVPR*, pages 1932–1939, 2009. 2

[8] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *CVPR*, volume 2, pages 142–149, 2000. 1

[9] V. Gazzola, G. Rizzolatti, B. Wicker, and C. Keysers. The anthropomorphic brain: the mirror neuron system responds to human and robotic actions. *Neuroimage*, 35(4):1674–1684, 2007. 2

[10] G. Guerra-Filho, C. Fermuller, and Y. Aloimonos. Discovering a language for human activity. In *Proceedings of the AAAI 2005 fall symposium on anticipatory cognitive embodied systems, Washington, DC*, 2005. 2

[11] B. Han, Y. Zhu, D. Comaniciu, and L. Davis. Visual tracking by continuous density propagation in sequential bayesian filtering framework. *PAMI, IEEE Transactions on*, 31(5):919–930, 2009. 2, 3

[12] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1296–1311, 2003. 1

[13] A. Kale, A. Sundaresan, A. Rajagopalan, N. Cuntoor, A. Roy-Chowdhury, V. Kruger, and R. Chellappa. Identification of humans using gait. *Image Processing, IEEE Transactions on*, 13(9):1163–1173, 2004. 2

[14] H. Kjellström, J. Romero, D. Martínez, and D. Kragić. Simultaneous visual recognition of manipulation actions and manipulated objects. *ECCV*, pages 336–349, 2008. 2

[15] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, pages 1817–1824. IEEE, 2011. 6

[16] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2):107–123, 2005. 2, 6

[17] C. Liu et al. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, MIT, 2009. 5

[18] E. Locke and G. Latham. *A theory of goal setting & task performance*. Prentice-Hall, Inc, 1990. 2

[19] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, pages 416–423, 2001. 4

[20] A. Mishra, Y. Aloimonos, and C. Fermuller. Active segmentation for robotics. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3133–3139. IEEE, 2009. 2, 3, 4

[21] T. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2):90–126, 2006. 2

[22] J. Neumann and etc. Localizing objects and actions in videos with the help of accompanying text. *Final Report, JHU summer workshop*, 2010. 6

[23] K. Nummiaro, E. Koller-Meier, L. Van Gool, et al. A color-based particle filter. In *First International Workshop on Generative-Model-Based Vision*, volume 2002, page 01, 2002. 5

[24] V. Papadourakis and A. Argyros. Multiple objects tracking in the presence of long-term occlusions. *Computer Vision and Image Understanding*, 114(7):835–846, 2010. 1

[25] G. Rizzolatti, L. Fogassi, and V. Gallese. Neurophysiological mechanisms underlying the understanding and imitation of action. *Nature Reviews Neuroscience*, 2(9):661–670, 2001. 2

[26] P. Saisan, G. Doretto, Y. Wu, and S. Soatto. Dynamic texture recognition. In *CVPR*, volume 2, pages II–58, 2001. 2

[27] M. Sridhar, A. Cohn, and D. Hogg. Learning functional object-categories from a relational spatio-temporal representation. In *ECAI*, pages 606–610, 2008. 2

[28] J. Tsotsos. Analyzing vision at the complexity level. *Behavioral and brain sciences*, 13(3):423–469, 1990. 3

[29] P. Turaga, R. Chellappa, V. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, 2008. 2

[30] I. Vicente, V. Kyrki, D. Kragic, and M. Larsson. Action recognition and understanding through motor primitives. *Advanced Robotics*, 21(15):1687–1707, 2007. 2

[31] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *ECCV*, pages 650–663, 2008. 2

[32] Y. Yang, M. Song, N. Li, J. Bu, and C. Chen. Visual attention analysis by pseudo gravitational field. In *ACM MM*, pages 553–556. ACM, 2009. 3

[33] B. Yao and L. Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *CVPR*, pages 17–24, 2010. 2

[34] A. Yilmaz and M. Shah. Actions sketch: A novel action representation. In *CVPR*, volume 1, pages 984–989, 2005. 2