To appear: IEEE International Conference on Pattern Recognition 2014

Interaction Recognition Using Sparse Portraits

Ivan Bogun*, Haidar Khan‡, Jacob Chen†, and Eraldo Ribeiro*
*Department of Computer Sciences, Florida Institute of Technology, Melbourne, Florida, U.S.A.
email: ibogun2010@my.fit.edu, eribeiro@fit.edu
†The University of Maryland, Department of Electrical and Computer Engineering, U.S.A.
‡Department of Electrical and Computer Engineering, The State University of New York at New Paltz, U.S.A.

***Abstract*—We propose a method for classifying actions involving people interacting with objects. Our method combines motion and appearance information into a unified framework. Here, we explore the video's sparse component as provided by robust principal-component analysis for the extraction of motion information in the form of trajectories. While we use motion as the main clue for classification, we also incorporate implicit object information into the classification process. Here, object information is represented by the probability of the object with which the person is interacting. These probabilities are learned using probabilistic Latent Semantic Analysis (pLSA). We test our classification method on a publicly available dataset, and provide a comparison with some related work. Classification results obtained by our method are promising.**

I. Introduction

The recognition of human actions is a key problem in computer vision [1]. An equally important but less explored problem is that of recognizing actions that involve people interacting with objects, which is the focus of this paper.

When we interact with objects, our actions are constrained by the object's properties such as shape and affordance [2], [3]. Combining information about both the person's motion and the object's appearance is a feature of successful algorithms for recognizing interactions. Indeed, while promising solutions to the problem have been proposed in the past 5 years [4], [5], many interesting questions remain unanswered including which feature should be used? Which parts of the interaction are relevant for recognition? How to detect such characteristic events in both space and time? For example, characteristic components of an interaction include the local image appearance when the person physically contacts the object and how that contact is performed. As pointed out by Filipovych and Ribeiro [6], [7], these *actor-object states* play a key role in interaction classification.

Trajectory-based methods gained attention recently with the work of [4], [5], [8]. Messing et al. [8] track keypoints to create a mixture of trajectory sequences. Activity is modeled as distributions over velocity history components, which results in a latent Dirichlet-allocation model. Gupta et al. [5] used upper-body pose detection to extract hand trajectories. Once the body was detected, the hand trajectory was segmented into components including reaching motion and manipulation motion. The method also accounted for object appearance and the object's reaction to interaction. This was done by considering changes in the object's state during interaction.

Prest et al. [4] performed interaction recognition in a weakly supervised manner (i.e., only the labels of the interactions were known). Similar to Gupta et al., they used a "person-first-then-object" approach by first detecting the person in the image by means of multiple detectors such as face, upper-body, full body. Then, objects that appeared frequently with respect to the person were located. Relative scale, distance, and overlap between the object and the person were used to compute human-object, whole-image, and pose-from-gradients descriptors. Classification was done using a multi-class SVM. State-of-the-art methods for recognizing interactions with objects require significant pre-processing steps like learning upper-body parts and object detectors [9], [10]. Also various descriptors used for learning are hard to interpret.

In this paper, we introduce a method that: (a) Requires only a head detector and does not explicitly use trackers. Instead, our method uses intuitive features including hand trajectory, hand velocity, and object appearance. (b) Uses hand trajectories obtained from the sparse component of the motion as the main source of information, and (c) Combines motion and appearance information into a multi-kernel SVM classification that uses implicit object information in the form of the probability of the object with it the person is interacting.

Recent developments in the convex-optimization area such as the Robust Principal Component Analysis (RPCA) [11] (or low-rank-plus-sparse decomposition) allow us to access important information from videos. Using RPCA, we can decompose videos into their moving (i.e., sparse component) and non-moving components (i.e., low-rank component). In our interaction-classification method, we explore the video's sparse component, as provided by RPCA, to detect characteristic moving trajectories and moment of interactions (Section III). The sparse motion component helps us locate the object in space so we can build a probabilistic model of the main object with which the person is more likely to be interacting. This model is learned using probabilistic Latent Semantic Analysis (pLSA) (Section IV). Once these models for both interaction motion and object appearance are at hand, classification is achieved using a multi-kernel SVM approach (Section V). In our method, we compare only parts of the trajectories where the interaction takes place. We assume that the time where the interaction starts and stops are known and leave the method which can extract those automatically for the future work. A description of our method is given next, and its feature-extraction step is illustrated in Figure 1.

II. Our Method

In our method, a video is seen as a 3-D spatio-temporal volume of N image frames. We commence by using RPCA to decompose each video into its low-rank and sparse components. This decomposition allows us to extract the fastest-

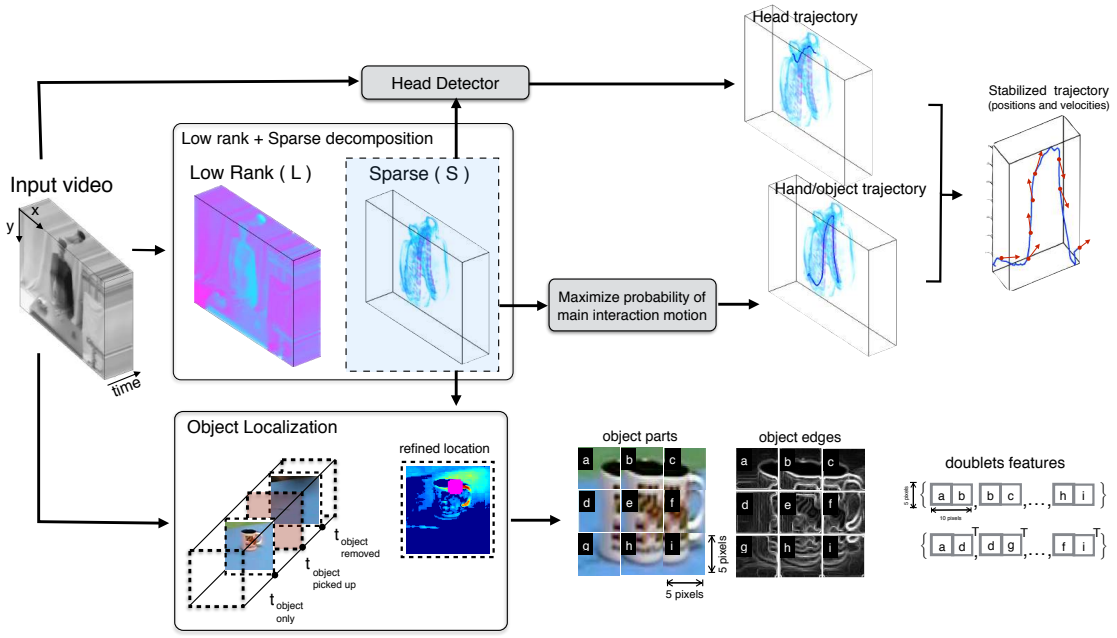

Fig. 1. Our method's feature-extraction pipeline. **Top**: trajectories and velocities are extracted from the sparse component of the video. **Bottom**: The object is localized and feature vectors representing its appearance are created.

moving regions in the video volume. We observed that in videos of a person interacting with an object (e.g., picking up an object from its original location and interacting with it), fast-moving regions tend to correspond to the interacting hand as well as the moving object. Because such motions occur in small regions of each video frame, the RPCA decomposition results in a stronger response in the sparse component than in its low-rank counterpart. The low-rank component will contain information related to the scene's background. Our method relies on the video's sparse component for locating and extracting information about both hand motion and object appearance. The interaction's sparse component can be visualized as a blurred spatio-temporal manifold (Left-hand side of Figure 2), which we name *the sparse portrait*.

We use both motion and appearance information for interaction classification. For the motion information, instead of tracking the hand or object directly in the video frames, we fit a trajectory by detecting at each time instant (i.e., video frame) a maximum-likelihood estimate over sparse portrait values and speed. The resulting curve is a good representation of the trajectory of the moving hand as well as the (moving) object. Appearance information is implicitly incorporated into the classification method in terms of the probability of an object. From trajectories, we learn one-vs-all support-vector machines for each interaction. The resulting classifications are weighted according to the probability of the object.

III. Motion Information

Our method uses motion information in the form of spatio-temporal trajectories that record the position of the target object over time. These trajectories are time-series of differing lengths, and can be obtained using a tracking-by-detection approach [12]. However, such methods can be sensitive to partial occlusions of the object and have their accuracy limited by the detector's. Instead, we capture motion trajectories directly from the sparse portrait.

A. The sparse portrait

We begin by extracting the video's fast-moving regions by using Robust Principal Component Analysis (RPCA) [11]. For this, we reshape each video frame as a row vector, and stack all of them into a matrix, X. RPCA solves the following convex-optimization problem:

$$\min . ||L||_* + \lambda ||S||_1 \quad \text{s.t.} \quad L + S = X, \tag{1}$$

where L is a low-rank matrix, S a sparse matrix, and λ is a parameter that controls the trade-off between sparsity and the rank of L. Under mild conditions, there is a range of λ where the recovery of matrices L and S is exact [11], i.e., it is possible to completely separate the video's moving regions from its static background. We observed that *the faster the object moves the more of an object appears in the sparse component*, thus we define a *sparse portrait* at the frame t as the matrix S_t. Examples of sparse portraits for the "drinking" interaction are given in Figure 2 (Left).

B. Trajectory extraction

We fit a set of trajectories to the sparse portrait by tracking locations that satisfy two conditions: they have the highest sparse portrait value and do not present significant changes in speed. Tracking of trajectory candidates is done by first detecting, in each frame, the location for which the S_t is maximum. Starting from these locations, tracking is performed

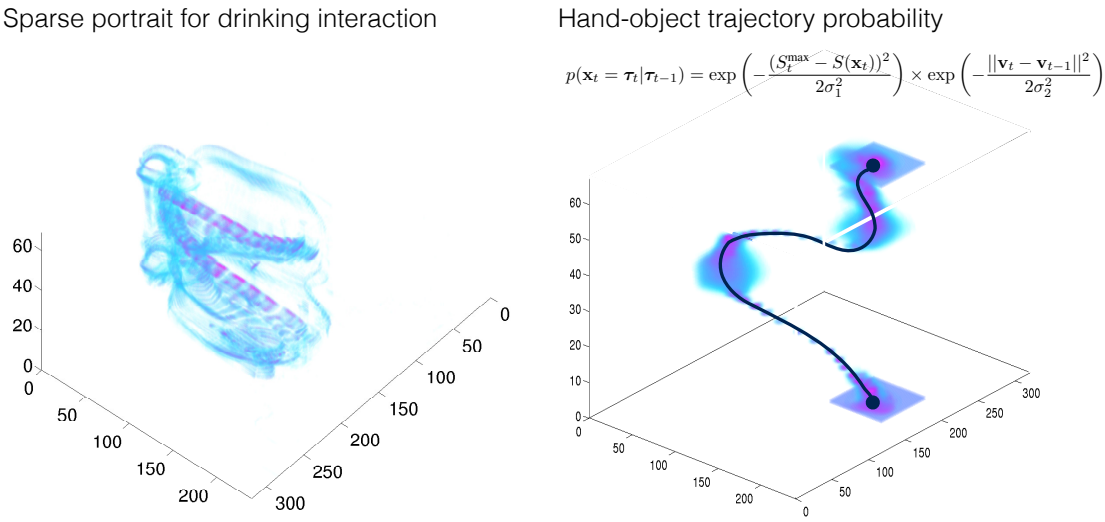

Fig. 2. **Left**: Sparse portraits volume for the drinking interaction. **Right**: Probability distribution of the main interaction motion, and fitted curve representing maximum likelihood trajectory of the main interaction motion.

both forwards and backwards in time. For numerical stability, each frame in the sparse portrait is smoothed with a Gaussian. Formally, let $\boldsymbol{\tau}_{t-1} = \mathbf{x}_{t-1} = (x_{t-1}, y_{t-1})$ be the previous position in trajectory $\mathcal{T}$. Also, let $\mathcal{N} = [i-r, i+r] \times [j-r, j+r]$ be the neighborhood of size r centered at $\boldsymbol{\tau}_{t-1}$. Let $S_t^{\max} = \max_{(i,j)\in\mathcal{N}} S_t$ be the maximum value of the sparse portrait, inside neighborhood $\mathcal{N}$, at time t. We define the probability of the current position of trajectory $\mathcal{T}$ as a function of velocity, $\mathbf{v}_t$, and sparse portrait value $S(\mathbf{x}_t)$:

$$p(\mathbf{x}_t = \boldsymbol{\tau}_t | \boldsymbol{\tau}_{t-1}) = \frac{1}{Z} \exp\left(-\frac{(S_t^{\max} - S(\mathbf{x}_t))^2}{2\sigma_1^2}\right) \times \exp\left(-\frac{||\mathbf{v}_t(\mathbf{x}_t) - \mathbf{v}_{t-1}||^2}{2\sigma_2^2}\right), \quad (2)$$

where σ_1 and σ_2 are parameters controlling smoothness, $\mathbf{v}_t(\mathbf{x}_t)$ is a velocity at $\mathbf{x}_t$ and $\frac{1}{Z}$ is a partition function. Once all trajectories are found, we choose the one whose sparse energy over all frames is the highest, i.e., the trajectory that traveled the most. Figure 2 (Right) shows the extracted trajectory for a drinking interaction.

We stabilize the trajectories by representing them with respect to a reference point, which we choose to be the head of the person performing the interaction. Here, for every frame, the head position is subtracted from the trajectory location in that frame. Then, in order to make it uncorrelated with the pixel distance, we divide each trajectory point by the largest distance from the head to the hand. To ensure that we could detect the head positions in every frame, we used the face detector by [13], with parameters that produced the most detections in each frame. Incorrect detection were subsequently filtered by removing the ones that were below the sparse-portrait's centroid, which is given by:

$$\vec{r}(t) = \frac{\sum_{i,j} \mathbb{1}_{\{S(\mathbf{x}_t)>0\}} (i,j)^\mathsf{T}}{\sum_{i,j} \mathbb{1}_{\{S(\mathbf{x}_t)>0\}}}. \quad (3)$$

Here, function $\mathbb{1}_{\{S(\mathbf{x}_t)>0\}}$ is the indicator function. This filtering procedure was effective because during the interactions the contour of person's silhouette always exhibited some motion that clearly appeared in the sparse portrait even for motions of small magnitude. Remaining head detections were tracked using the tracker by Kalal et al. [12], resulting in a trajectory. We further filtered out trajectories whose the total sparse-portrait values (i.e., integrated over the trajectory's length) were below a predefined threshold. Finally, the trajectory that overlapped the most with the others was selected as the final. Note that, in the videos we analyzed, while head trajectories were mostly static, they were not constant.

Velocities play an important role in the dynamics of the interaction, and we include them as features in our classification method. For example, a brief pause for dialing a number into a mobile phone is reflected in the trajectory's velocity vectors. We calculate velocities using pairwise differences, i.e., $\mathbf{v}_t = \boldsymbol{\tau}_t - \boldsymbol{\tau}_{t-1}$. The motion information used by our classification method is then given by:

$$\mathbf{m} = \{(\boldsymbol{\tau}_1, \mathbf{v}_1), \ldots, (\boldsymbol{\tau}_n, \mathbf{v}_n)\}. \quad (4)$$

IV. Appearance Information

In addition to motion information, knowing something about the object with which the person is interacting can help classification, especially when different interactions undergo similar motions. In our method, we do not use object appearance explicitly. Instead, we need only to know the probability that a given object is being moved by the person. To model this probability, we use probabilistic latent semantic analysis (pLSA) with a Bag-Of-Words (BoW) appearance representation similar to Sivic et al. [14].

Given an image of the object, we represent its appearance by two sets of features: edge magnitudes obtained from the Canny detector, and gray-level intensities. The two types of

images are subdivided into nine 5-pixel square patches. We call the edge-map features *singles*. These singles are stacked together to form a 25-length feature vector, $\mathbf{a}_s$. To account for spatial information, we combine the pairs of neighboring patches in the intensity images along the horizontal and vertical directions. We call these features *doublets*. The doublets are reshaped to form 50-length feature vectors, $\mathbf{a}_d$. Figure 1 (bottom-right) illustrates how doublets are formed.

Using these patch-based vectors can inevitably result in an insufficient number of features when constructing the BoW dictionary. To address this issue, we collect features from multiple video frames containing the object.

A. Probabilistic Latent Semantic Analysis

Once the appearance vectors for the object are computed, the probability of an object being used for an interaction can be learned using the EM algorithm [15]. Here, we assume that each video has multiple features and a unique latent label which is the object's identity. Let $\mathcal{V}_i$ denote video i, $\mathbf{a}_j$ the feature j, z_i the label of the video i. We initialize $P(z_k|\mathcal{V}_i)$ randomly. The following EM formulation is taken from Hofmann (2001) [15]:

a) E-step:

$$P(z_k|\mathcal{V}_i, \mathbf{a}_j) = \frac{P(\mathbf{a}_j|z_k)P(z_k|\mathcal{V}_i)}{\sum_{l=1}^{K} P(\mathbf{a}_j|z_l)P(z_l|\mathcal{V}_i)}. \quad (5)$$

For numerical stability, we added Laplace smoothing in $P(\mathbf{a}_j|z_k)$. Let us denote:

$$p_j(z_k) = \sum_{i=1}^{N} P(\mathcal{V}_i, \mathbf{a}_j)P(z_k|\mathcal{V}_i, \mathbf{a}_j) + \gamma_j. \quad (6)$$

b) M-step:

$$P(\mathbf{a}_j|z_k) = \frac{p_j(z_k)}{\sum_{j=1}^{M} p_j(z_k)}. \quad (7)$$

$$P(z_k|\mathcal{V}_i) = \frac{\sum_{j=1}^{M} P(\mathcal{V}_i, \mathbf{a}_j)P(z_k \mid \mathcal{V}_i, \mathbf{a}_j)}{P(\mathcal{V}_i)} \quad (8)$$

In our experiments we used $\gamma_j = 1, \forall j$.

The distribution $P(z_k|\mathcal{V}_i)$ gives a probability of object k being used for the interaction in video i. Knowing the probability of the object used in an interaction can significantly restrict the subset of possible interactions during classification. Numerically, we propagate this information by weighting the classification result obtained from trajectories and velocities with probabilities of the objects (Section V).

B. Object Localization

Our learning algorithm is weakly supervised, i.e., we know the type of interaction but we do not provide annotated locations of the object. However, to train the pLSA model we need to locate the object in the video frames to extract its appearance features. Our solution is to use the sparse portrait to try to detect both the location and the instant when the object is removed from its original position. For the videos used in our experiments, this happens when the objects are being picked up from the top of the table.

Starting from the first frame of the interaction, we find the lowest location with respect to the body position (i.e., location $\vec{r}(t)$ given by Equation 3), for which the sparse energy is the highest. Such position in the sparse portrait will correspond to the motion when the person was picking up the object from the table. Because this location is only an approximation of the object's position, we refine it by extracting image patches around that location at a time when the object was not yet grasped (i.e., the object is still on the table) and also at the time when the object was no longer there (i.e., during the interaction). Because the initial location is not the exact location of the object, we must extract image regions that are large enough not to miss the object. Figure 3 (a) shows patches extracted around the initial localization during the time when the object was not yet touched. Figure 3 (b) shows patches extracted after the object was picked up.

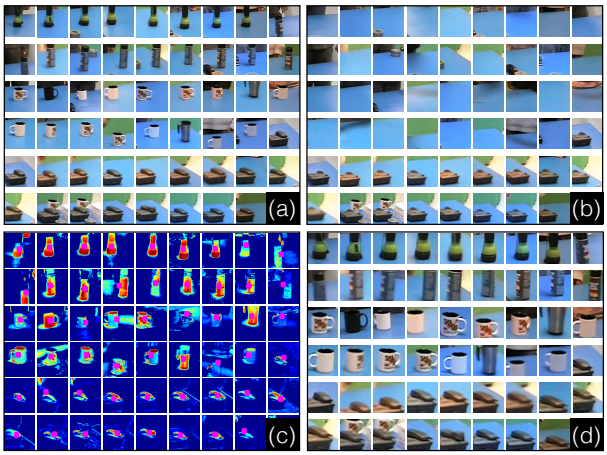

Fig. 3. Detecting the object that is target of the interaction. (a) Object is still on its original location. (b) Object has been removed from its original location. (c) Difference image with refined location of the object (in magenta). (d) Region containing the object, centered at the refined location.

We refine the initial object location by computing the centroid of the difference image between the two patches in the LMS color space [16]. Because only the object was removed from the image patch, the centroid is the mean position of the difference between the patches, which is very likely to be the object's center. Figure 3 (c) shows the image difference and the refined object's center (i.e., large cyan dot). Figure 3 (d) shows new image regions extracted at the refined locations.

V. Kernel-Based Classification

Trajectories, velocities, and object appearance are complementary information about the underlying interaction. We use these three characteristics to define kernels that measure the similarity between two videos. Once the kernels are constructed, we train support-vector machines classifiers using a one-vs-all scheme for multi-class classification (i.e., a separate binary SVM classifier was learnt for each interaction type). While our appearance vectors are of fixed size, our motion trajectories are time series of different lengths. To address this

issue, we adopt the dynamic-time alignment kernel from [17], which can align and compare times series of different lengths.

A. Time Series Alignment Kernel

Consider two time series or sequences, $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ and $\mathbf{q} = (q_1, q_2, \ldots, q_m)$ of lengths n and m, respectively. Let $\mathcal{A}(\mathbf{s}, \mathbf{q})$ be the set of all possible alignments, π, between sequence $\mathbf{s}$ and sequence $\mathbf{q}$ where $|\pi| = p$. The alignments π_1 and π_2 are mappings from indices of sequences $\mathbf{s}$ and $\mathbf{q}$ to $1, \ldots, p$. Essentially, mappings π_1 and π_2 provide a way to compare each element in $\mathbf{s}$ with an element in $\mathbf{q}$. The alignment kernel is given by:

$$\mathcal{K}_1(\mathbf{s}, \mathbf{q}) = \underset{\pi \epsilon \mathcal{A}(\mathbf{s},\mathbf{q})}{\operatorname{argmax}} \frac{1}{|\pi|} \sum_{i=1}^{|\pi|} \exp\left(-\frac{||s_{\pi_1(i)} - q_{\pi_2(i)}||^2}{\sigma^2}\right). \tag{9}$$

Other kernel methods based on dynamic time warping exist [18] and could also be used. The kernel in Equation 9 can be computed in $O((\max\{n, m\})^2)$ time. To avoid scaling differences, all kernels were normalized as suggested by [19] using a normalization transformation that preserves the kernel's positive semi-definiteness.

B. Classification

Now that the motion-similarity kernel is defined, we incorporate the probability of object into the classification process. First, we use the alignment kernel to learn one-vs-all SVMs over trajectories and velocities separately. Let D^t and $D^v \in \mathbb{R}^{d \times c}$ be the resulting classifications obtained for trajectories and velocities, respectively, where d is the number of data points in the dataset while c is the number of classes.

In order to propagate object similarity into the motion-based classifications, we weight D^t and D^v with the probability of the object. Formally, we have:

$$D^t_w = \begin{cases} D^t(\mathcal{V}_i, c_j) p_{\text{obj}} & \text{if } D^t(\mathcal{V}_i, c_j) \geq 0 \\ D^t(\mathcal{V}_i, c_j)(1 - p_{\text{obj}}) & \text{otherwise,} \end{cases} \tag{10}$$

where $p_{\text{obj}} = P(z_j|\mathcal{V}_i)$ is the probability of object from class z_j from video $\mathcal{V}_i$. D^v_w is defined similarly. The final classification is performed using:

$$L(\mathcal{V}_i) = \arg\max_{c_j} \left\{ \max\left(D^t_w, D^v_w\right) \right\}. \tag{11}$$

VI. Experiments

We evaluated our method on the Gupta's dataset [5]. The dataset consists of 54 videos of individuals performing interactions with objects. Interactions include spraying, answering the phone, and lighting a flash light. At first, we computed sparse plus low-rank decomposition for every video using the TFOCS toolbox[1]. Then, we extracted hand/object trajectories as described in Section III-B, and setting $\sigma_1 = 12$ and $\sigma_2 = 8$. Head trajectories were extracted using a pre-learnt model from the face-detection method in [13][2]. For head trajectories, we also used the tracking algorithm by [12] with its default parameters[3]. We normalized trajectories with respect to the head position and divided by the norm of the maximum distance along the hand/object trajectory. Velocities were calculated as pairwise differences of consecutive trajectory points. The plot in Figure 4 shows the average pixel error per video per frame for trajectory extraction in comparison with manually extracted trajectories, which we consider as ground truth. Clearly, some trajectories are not extracted correctly and cause misclassification.

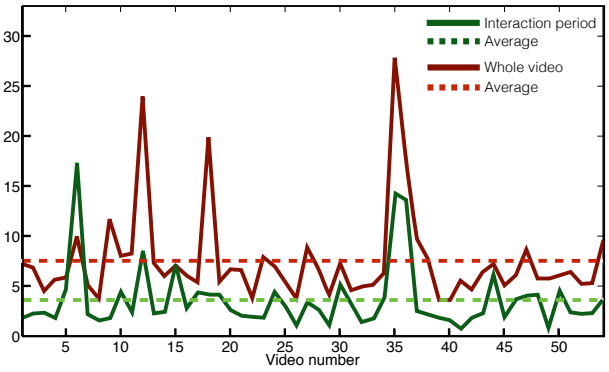

Fig. 4. Trajectory extraction error. Average pixel-distance error per frame of the estimated trajectory with respect to the (manually annotated) ground-truth trajectory of the person's hand. The green solid line represents the trajectory error for time interval when the interaction took place while red solid line is the error for the entire video. Corresponding single average values for both measurements are shown in the horizontal dashed lines of similar lighter color.

Due to limited nature of the dataset, we used a leave-one-out cross-validation. For object localization, we initially extracted a region of interes larger than the expected size of the object with width and height of 100 pixels. Refinement of the initial localization was performed as described in Section IV-B. The final object patch was extracted with a 60-pixel square region of interest. For each video, we extract patches for the first n consecutive frames. We observed that the more frames we used the better was the classification and around $n = 10$ it leveled off. Singles and doublets features were computed to create two dictionaries describing the appearance of the object. Each dictionary were created using k-means clustering with $k = 55$. Then, we applied pLSA on edges using random initialization, after convergence we fed $P(z|d)$ from edges as an initialization to doublets after which pLSA was run again. The results for object classification using our method are given in Table I. As described in Section V-B, the probability distributions of the objects were used to weight the results of classification obtained from trajectory and velocities SVMs. The final classification of the interactions as given by Equation 11 are shown in Table II.

TABLE I. Object recognition results

Method	Object recognition accuracy %
Doublets to singles	84.6 ± 2.2

[1] http://cvxr.com/tfocs/

[2] code available at http://www.ics.uci.edu/~xzhu/face/

[3] http://personal.ee.surrey.ac.uk/Personal/Z.Kalal/tld.html

TABLE II. CLASSIFICATION RESULTS ON THE GUPTA DATASET. FOR ALL SVMS, WE FIXED $C = 1, \sigma = 100$.

	Method	Accuracy (%)
Our method	Trajectories only	50.0
	Velocities only	59.3
	Trajectories+object	70.0 $\pm$ 1.1
	Velocities+object	72.4 $\pm$ 1.8
	Full framework	82.2 $\pm$ 2.3
Related work	Gupta et al. [5]	93.0
	Prest et al. [9]	93.0

Our method has a number of drawbacks. First, we did not address the problem of automatically localizing the interaction in time. Some previous work have looked at this problem [20], [21]. Secondly, classification quality is linked to the quality of the trajectories. Incorrect trajectories will corrupt both velocity information and object appearance.

In the work reported in this paper we extracted trajectories using only *motion* information. As a result, trajectories of objects that temporarily stop moving might be incorrectly extracted. One way to address this issue is to combine tracking in sparse portraits with state-of-the-art trackers that operate on images. In addition to that, it became apparent that there is a lack of object detectors that can work in *videos* as objects move (See [4]). Because object motion causes their imaged shape to change, appearance-based object detectors that may work for static images do not directly extend to videos.

VII. CONCLUSIONS

In this paper, we presented a method for interaction recognition that operates in weakly supervised settings. Using trajectories as the main source of information, we show how to extract them from the sparse component of the video, which we extract using robust PCA. Except for a head detector, our method does not use specialized trackers or detectors. Instead of using object appearance directly, we incorporate the probability of object appearance into a multi-kernel classification scheme. While our classifications are still below state-of-the-art methods like the one by Gupta et al. [5], we consider our results promising given the weakly supervised natured of our method. Future work include improving tracking of fast motions by incorporating measurements from sparse components of the video into image-based tracking methods.

This work was partially supported from National Science Foundation (NSF) grant No. 1263011.

REFERENCES

[1] R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing*, vol. 28, no. 6, pp. 976–990, 2010.

[2] H. Kjellström, J. Romero, and D. Kragić, "Visual object-action recognition: Inferring object affordances from human demonstration," *Computer Vision and Image Understanding*, vol. 115, no. 1, pp. 81–90, 2011.

[3] A. Gijsberts, T. Tommasi, G. Metta, and B. Caputo, "Object recognition using visuo-affordance maps," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 1572–1578.

[4] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari, "Learning object class detectors from weakly annotated video," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3282–3289.

[5] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: Using spatial and functional compatibility for recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 10, pp. 1775–1789, 2009.

[6] R. Filipovych and E. Ribeiro, "Robust sequence alignment for actor–object interaction recognition: Discovering actor–object states," *Computer Vision and Image Understanding*, vol. 115, no. 2, pp. 177–193, 2011.

[7] ——, "Recognizing primitive interactions by exploring actor-object states," in *CVPR*, 2008, pp. 1–7.

[8] R. Messing, C. Pal, and H. Kautz, "Activity recognition using the velocity histories of tracked keypoints," in *ICCV '09: Proceedings of the Twelfth IEEE International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2009.

[9] A. Prest, V. Ferrari, and C. Schmid, "Explicit modeling of human-object interactions in realistic videos," INRIA, Technical Report RT-0411, Sep. 2011. [Online]. Available: http://hal.inria.fr/inria-00626929

[10] A. Prest, C. Schmid, and V. Ferrari, "Weakly supervised learning of interactions between humans and objects," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 3, pp. 601–614, 2012.

[11] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, p. 11, 2011.

[12] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 7, pp. 1409–1422, 2012.

[13] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2879–2886.

[14] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1, 2005, pp. 370–377. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1541280

[15] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Machine learning*, vol. 42, no. 1-2, pp. 177–196, 2001.

[16] M. D. Fairchild, *Color appearance models*. Addison-Wesley-Longman, 1997.

[17] H. Shimodaira, K.-i. Noma, M. Nakai, and S. Sagayama, "Dynamic time-alignment kernel in support vector machine," *Advances in Neural Information Processing Systems (NIPS2002)*, no. 921-928, 2002.

[18] M. Cuturi, J.-P. Vert, O. Birkenes, and T. Matsui, "A kernel for time series based on global alignments," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2. IEEE, 2007, pp. II–413.

[19] A. B. A. Graf, A. J. Smola, and S. Borer, "Classification in a normalized feature space using support vector machines," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 597–605, 2003. [Online]. Available: http://dblp.uni-trier.de/db/journals/tnn/tnn14.html#GrafSB03

[20] A. Kläser, M. Marszałek, C. Schmid, and A. Zisserman, "Human focused action localization in video," in *Trends and Topics in Computer Vision*. Springer, 2012, pp. 219–233.

[21] I. Laptev and P. Pérez, "Retrieving actions in movies," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.