Interactive Action Recognition from 3D Skeleton Data

Swami Sankaranarayanan, Kota Hara, Yiannis Aloimonos
Center for Automation Research, University of Maryland, College Park, MD 20742
{swamiviv,kotahara}@umiacs.umd.edu

Abstract

In this work, we propose

1. Introduction

Human action recognition has been a major problem in computer vision community for many years. There are many applications of the human action recognition such as surveillance, health-care and human computer interaction. Most of the previous works focus on action recognition of a single person. The action classes considered in this scenario is limited to a simple action such as walking, running and jumping.

On the other hand, there are several works which deal with action classes involved with multiple people. These action classes are more complex to recognize than the single-person actions due to the complexity of the actions. The actions are defined not only by the motion of one person but also their interactive motions. We call these multiple-people actions as interactive actions.

The examples of interactive actions we consider in this work are exchanging objects and greetings between two people. For exchanging objects, we consider different types of the objects such as cards, a ball and a chair. Depending on the object being exchanged, their interactive movement come out in a different form. For greetings, we consider interactive actions such as shaking hands and bowing. Our goal is to recognize these different types of interactive actions from the 3d skeleton information.

Action recognition from the body skeleton information has been actively studied due to the advance of motion capture systems such as Kinect. Typically the skeleton information is given in the form of 3d locations of a set of body joints. Many action recognition systems directly use these information as the input or convert the location information into a set of joint angles and then use them as the input. The focus of most of the previous research has been on the recognition part.

[1] proposed a new skeleton representation based on relative geometry across body parts and achieve the state of the art results on several standard single-person action recognition datasets. They compute the translation and rotation between each pair of body parts and map it to the tangent space to obtain a feature vector. The benefit of their approach compared to the previously adopted skeleton representation is that their representation can directly capture the relationship between body parts that are not directly connected each other. Thus, it is more capable of recognizing actions in which the relationship between non-connected body parts, such as left arm and right arm, is more important.

In this work, we extend [1] to classify actions performed by two people. In our skeleton representation, in addition to the relative geometry across body parts within one person, we also consider relative geometry between body parts across two people. We believe that this representation enables us to capture two people's interaction well. For instance, our representation can captures the geometric relationship between a left arm of a person A and a right arm of a person B, which might be helpful to recognize the 'shaking hands' actions.

For the recognition part, we develop an algorithm which not only predicts the action class but also localizes frames where actions are taking place. This is done by using Latent Structured SVM framework. Specifically, we treat the staring and ending frames of the action as latent variables and jointly optimize the cost function in a max-margin manner. Note that this method does not require annotation for latent variables as they are automatically determined in the optimization process.

The paper is organized as follows. In section 2, we show some related work. Section 3 explains our action localization method. Then we describe, in section 4, the proposed skeleton representation at each frame. Section 5 presents the interpolation method to obtain a final feature vector as well as the classification methods we propose. The experimental results are shown in section 6, followed by the conclusion in section 7.

2. Related Work

3. Heuristic Approach for Action Localization

The action sequences that are collected from the Kinect consists of several irrelevant frames which does not contain the action that we are looking for. Thus, given a video sequence it becomes important to approximately localize the action. This is a mandatory pre-processing step to extract realiable features that completely describe the action that we are interested in. We alos assume throughout this work that there is only one continuous action that takes place throughout a given video sequence. In this section, we decribe a heuristic way to localize the action in a given video. A given video sequence consists of the following parts:

$$< Init.Action - MainAction - Finish > \quad (1)$$

The heuristic described here tries to acquire the frames corresponding to the Main Action in an automated way. We perform the localization in two steps:

- Computing the action center frames
- Computing the start and end frames

The action center frame is computed as the frame where the distance between the interacting persons is the least. If there are multiple frames with the same distance value (this could occur since some interactions can last for a few frames), the action center frame is assigned as the minimum of those frames.

The start(end) of any interaction in the video could be implied by a constant decrease(increase) in distance between two persons. Thus, we compute the distance between the persons at each frame and identify when the rate of change of distance goes above(start) or below (end) a given threshold δ, which is a user input parameter.

As can be observed, this method does not yield us exact localization but an approximate one. In the subsequent section, we take care of this by imposing temporal smoothness on the features that we extract from each frame.

4. Skeleton Representation

The output from the Kinect sensor is 3d positions of 20 body joints for each person defined in the global coordinate. We define our skeleton representation for the first person as $S^{(1)} = (V^{(1)}, E^{(1)})$, where $V^{(1)} = \{v_1^{(1)}, v_2^{(1)}, \dots, v_N^{(1)}\}, N = 20$ denotes the set of joints and $E^{(1)} = \{e_1^{(1)}, e_2^{(1)}, \dots, e_M^{(1)}\}, M = 19$ denotes the set of body parts (See Fig.1). Let $e_{m1}^{(1)}, e_{m2}^{(1)} \in \mathcal{R}^3$ represents the starting and ending points of the body part $e_m^{(1)}$. Similarly we have $S^{(2)} = (V^{(2)}, E^{(2)})$ for the second person.

First, we define the person specific local coordinate at the first person's shoulder center computed as $(v_5 + v_6)/2$ such that the x coordinate is aligned with the person's body orientation. Then we update both $S^{(1)} = (V^{(1)}, E^{(1)})$ and $S^{(2)} = (V^{(2)}, E^{(2)})$ using this new coordinates. This process makes our representation invariant to the global translation and orientation.

Next we consider a set of body parts, $E = \{E^{(1)}, E^{(2)}\}$, and for each pair of body parts e_m and e_n, in E, we describe their relative geometry by the translation and rotation. The translation is computed as $T_{m,n} = e_{m1} - e_{n1}$. For the rotation, we compute the rotation axis r and the rotation angle θ between two vectors $e_{n2} - e_{n1}$ and $e_{m2} - e_{m1}$. From them we compute a quaternion $q_{m,n} \in \mathcal{R}^4$ by $q_1 = r_1 sin(\frac{\theta}{2}), q_2 = r_2 sin(\frac{\theta}{2}), q_3 = r_3 sin(\frac{\theta}{2}), q_4 = cin(\frac{\theta}{2})$.

Thus, each pair of parts can be represented as 7 dimensional vector. Since there are $\binom{38}{2}$ such pairs, we have $\binom{38}{2} \times 7 = 4921$ dimensional vector. We do the same step by using the second person's local coordinate and concatenate two vectors to obtain the final feature vector representing two people configurations at a current frame.

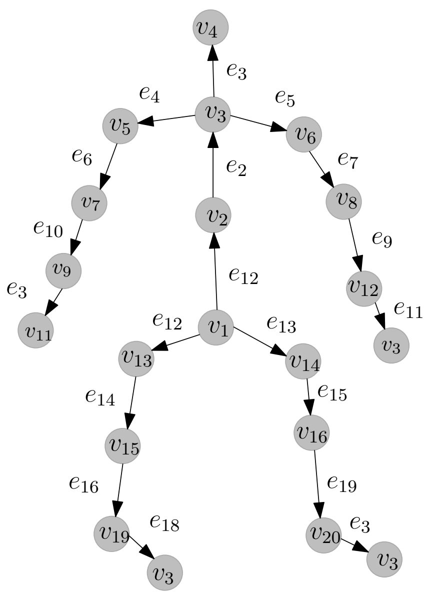

Figure 1: An illustration of the skeleton.

We compute our skeleton feature across 41 frames which includes the action center frame, 20 frames before the action center frame and 20 frames after the action center frame. Finally we concatenate all the feature vectors into a final vector.

5. Joint Action Localization and Classification

In Sec.3, we proposed a heuristic approach for action localization where the aim is to automatically determine when the action starts and ends. The problem of this approach is that it is not designed to achieve better classification accuracy. Also it is purely subjective to define the start and the end of the action.

Thus, we propose a method which can do both tasks simultaneously. In the proposed method, the training does not require manual labeling of the starting and ending frame of each action sequence. The model is trained in such a way that the starting and ending frame are determined in order to maximize the discriminativeness of the model.

Our method is based on a latent structured SVM framework [6]. In this framework, we are given a set of training data

$$\mathcal{S} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \in (\mathcal{X}, \mathcal{Y})^N$$

, where in our case, x is skeleton feature extracted from a whole sequence and y is an action class label. Let h denote a latent variable, which specifies the starting and ending frame of the sequence.

The prediction is done by finding h and y which maximize the following scoring function:

$$f_{\mathbf{w}}(x) = \underset{(y,h)\in\mathcal{Y}\times\mathcal{H}}{\operatorname{argmax}} [\mathbf{w} \cdot \Phi(x, y, h)] \quad (2)$$

where $\Phi(x, y, h)$ is a joint feature map. As a result, we not only obtain the predicted action class but also the starting and ending frame of the action in the given sequence.

In this work, we define the joint feature map $\Phi(x, y, h)$ as follows.

- (Step 1) Extract part of x between starting frame and ending frame (denoted as $\hat{x}$)
- (Step 2) Expanding/Shrinking $\hat{x}$ by temporal resampling so that the dimensionality becomes a predefined number (denoted as x^*)
- (Step 3) Setting y-th section of Φ as x^* and remaining part as 0

$$\Phi = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ x^* \\ \vdots \\ 0 \end{pmatrix} \in R^{|\mathcal{Y}| \times dim(x^*)} \quad (3)$$

The last step is a common approach taken for multiclass classification problems. This way, for each action class, different part of $\mathbf{w}$ is used and the class which has the highest score is chosen by Eq.2.

The training is done by solving the following optimization problem:

$$\min_{\mathbf{w}} ||\mathbf{w}||^2 + C\sum_{i=1}^{N}[\max_{(\tilde{y},\tilde{h})\in(\mathcal{Y},\mathcal{H})}[\Delta(y_i,\tilde{y}) + \mathbf{w}\cdot\Phi(x_i,\tilde{y},\tilde{h})] - \max_{h\in\mathcal{H}} \mathbf{w}\cdot\Phi(x_i, y_i, h)] \quad (4)$$

where $\Delta(y_i, y_j)$ is a loss function. We define the loss as

$$\Delta(y_i, y_j) = \begin{cases} 0 & (y_i = y_j) \\ 1 & (otherwise) \end{cases}$$

By rewriting Eq.4,

$$\min_{\mathbf{w}} ||\mathbf{w}||^2 + C\sum_{i=1}^{N}[\max_{(\tilde{y},\tilde{h})\in(\mathcal{Y},\mathcal{H})}[\Delta(y_i,\tilde{y}) + \mathbf{w}\cdot\Phi(x_i,\tilde{y},\tilde{h})]] - C\sum_{i=1}^{N}[\max_{h\in\mathcal{H}} \mathbf{w}\cdot\Phi(x_i, y_i, h)]$$

it is clear that the objective function is non-convex as it is defined as the difference between two convex function. We thus find a local optimum by EM-like algorithm where we first find, for each training sample, h which maximize $\mathbf{w} \cdot \Phi(x_i, y_i, h)$. In our setting, this corresponds to finding the starting and ending frame for a training sample which make the score as high as possible.

In the second step, we fix h and minimize $||\mathbf{w}||^2 + C\sum_{i=1}^{N}[\max_{(\tilde{y},\tilde{h})\in(\mathcal{Y},\mathcal{H})}[\Delta(y_i,\tilde{y}) + \mathbf{w}\cdot\Phi(x_i,\tilde{y},\tilde{h})]]$, which is convex. The maximization in the objective function corresponds to finding the most strict constrain for the optimization problem. To put it simply, we find a class ($\tilde{y}$ and h where ($\tilde{y}$ is different from the true class but the score is high.

The training is done by iteratively executing the above two steps while the decrease of the objective value becomes marginal.

Though we can use random values for the initial values for h, it is often desired to give a better initial values for h. We use the results of the heuristic approach described in Sec.3 as our initial values for h.

6. Experiments

6.1. Data Collection

We create a new dataset named Maryland Interactive Action Dataset (MIAD) which consists of 10 interactive action classes captured by Kinect. The action classes we consider are summarized in Fig.2 with their representative images. For each pair of people, we collect 2 sequences per an action class by switching their locations. Since we collect data from 6 pairs of people, there are 12 sequences per action

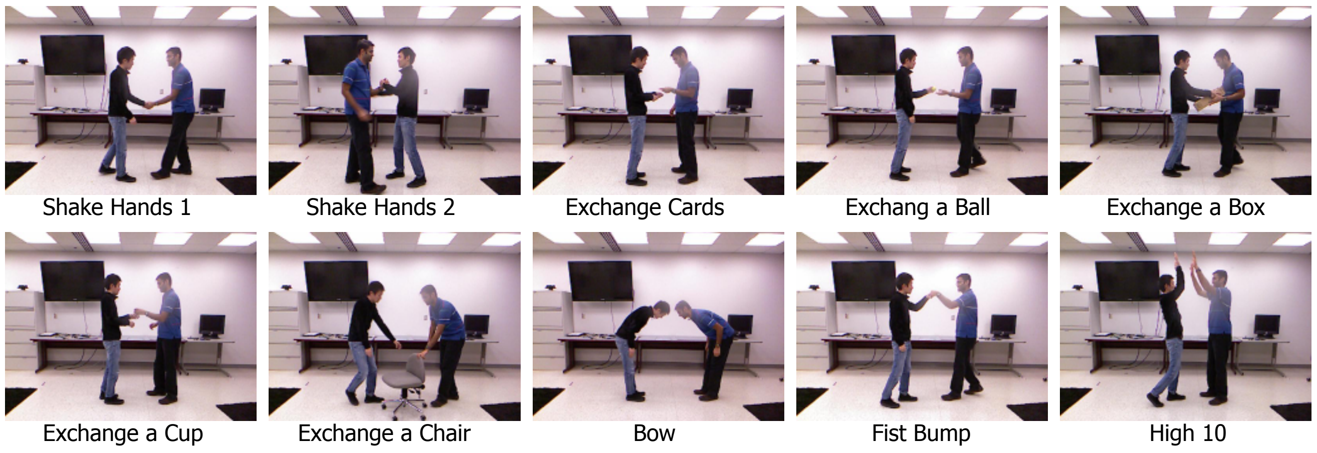

Figure 2: Action classes in newly collected MIAD dataset.

class. We use the first X sequences as training data and the remaining sequences as testing data.

Our new data, MIAD differs from the existing two people interactive dataset such as K3HI [3] and SBU Kinect Interaction Dataset [4]. First, our skeleton is represented by 20 joints whereas K3HI and SBU have only 15 joints. Second, the action classes we consider are more 'fine-grained'. In K3HI and SBU, there is only one action class for 'exchanging an object' class while MIAD contains 5 unique action classes for it, differentiated by the object being exchanged. Also K3HI and SBU have only one action class for the greeting action, namely, shaking hands, while MIAD includes 5 different action classes. These fine-grained actions are generally more challenging to discriminate as actions are more similar each other, necessitating more detailed representation of body movements.

6.2. Action Localization Results

In this section we show some sample results of our Action localization procedure described in Section **??**. Figure 3 shows the difference in the distance profiles be-

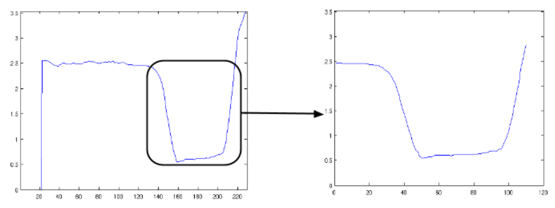

Figure 3: Distance Profile before(left) and after(right) localization

tween the raw action sequence and the processed action sequence. As can be seen from Figure 3, the processed distance profile shows an appromiately zoomed in portion of the original profile. This localized region represents the frames in the given Video as to where the interaction takes place.

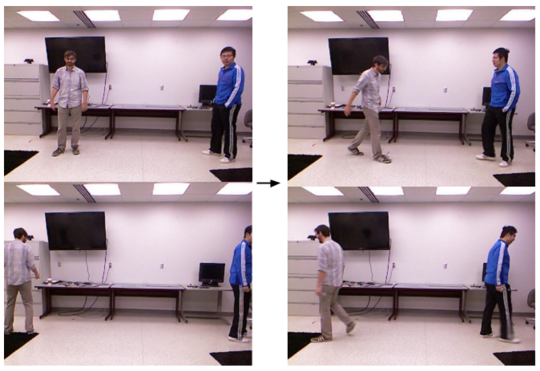

Figure 4: Start(Top) and End(Bottom) Frames

Figure 4 shows the start and end frames corresponding to the above action profiles. Thus, we use this simple heuristic method to extract only the relevant portion of the action sequence and provide this as input to our Feature extraction operator.

6.3. Action Classification Results

To classify a given test input to one of the action classes, we learn a discriminative mapping between the input space and the output action label space. In the training stage, we learn the parameters of this mapping/classifier and apply them to classify the test input. In this work, we have used the following classification methods:

- Support Vector Machines (SVM): one (vs) one, one (vs) all
- k- Nearest Neighbour: k = 1,4

- Classification Trees

SVM's are desgined to handle binary classification problems. To use them for multi-class classification, we use two paradigms which are explained as follows:

- **1(vs)1**: In this case, for each class combination (i, j) one binary classifier is learnt. Thus for a N class problem, the total number of classifiers becomes $\frac{N(N-1)}{2}$.
- **1(vs)all**: In this case, for each class i in the training set, one classifier is learnt. The positive examples for this classifier are the inputs from class i while the ngative examples are the rest of the inputs from all other classes. Thus, N classifiers are learnt, one per class.

Since SVM's produce only the decision values for a multi-class setting, we use Platt scaling as described in [5] to convert these decision values to valid posterior probability estimates. Thus, for each classifier, for each test input x_t, we obtain the posterior probability measure $p(y = i|x_t)$, where i is any class label between $\{1, 10\}$ and assign that i to x_t which maximizes the probability measure. For each of these methods, we use different experimental settings by using different amounts of training and test data for each setting. The parameters for SVM are tuned using cross-validation on the training set.

Table 1: Classification Accuracy

Split-Ratio	SVM(1-1)	(1-all)	knn-1	knn-4	CT
20:80	18.4	29.5	21.4	11.2	23.4
40:60	17.4	33.3	26.1	18.8	23.1
60:40	26.5	40.8	32.7	20.4	**28.6**
80:20	**40**	**50**	**40**	**40**	25

The results for our classification method are shown in Table 1. Figure 5 shows the confusion matrix for the best performing classification method. The learning curve for the best performing classifier is shown in Figure 6.

Discussion

From the confusion matrix in Figure 5, we can observe that the most confusing actions correspond to labels 1, 2 and 5, 6. Both labels 1, 2 correspond to variations of the shaking hands action, one being the normal way and the other being an upright shake hands action. The scenario is shown in Figure 7. Given that these actions are very difficult to distinguish and that different people perform these actions in different ways (such that the disriminative aspect is lost), the features extracted from the limited training data that we have could not clearly distinguish between these actions.

Labels 4, 5, 6 correspond to exchanging objects; box, ball and cup respectively. These could be more accurately classified by giving an additional information to the classifier involving the description of the object that appears in these frames. This part is a work under progress where we are trying to find an effective way to obtain an object description that would not be very computationally expensive. Furthermore, the learning curve in Figure 6 shows that our model has very high variance, since the test error is much higher than the error on the training set. This implies that we can bridge the gap and thus improve the accuracy of our method by using more training data.

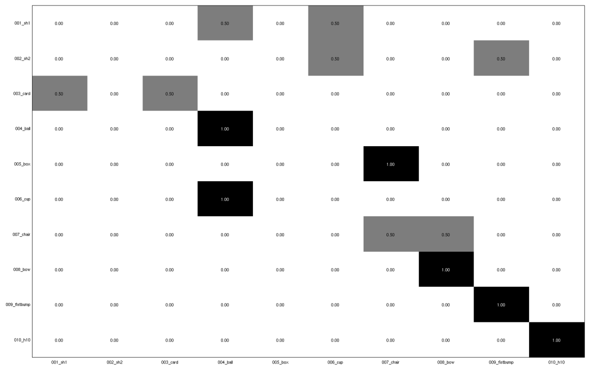

Figure 5: Confusion matrix for the best performing classification method (SVM one-vs-all). Darker color indicates larger values

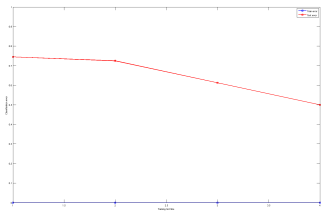

Figure 6: Learning Curve for SVM one-vs-all method

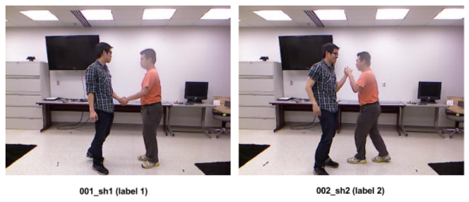

Figure 7: Variants of the shake hands action (Labels 1 and 2)

7. Conclusion

References

[1] Raviteja Vemulapalli, Felipe Arrate, Rama Chellappa, *Human Action Recognition by Representing 3D Skeletons as Points in a Lie Group*. CVPR, 2014. 1

[2] Semantic-level Understanding of Human Actions and Interactions using Event Hierarchy, Sangho Park, J.K. Aggarwal, CVPR'04

[3] Sangho Park, J.K. Aggarwal, *K3HI: Kinect-based 3D Human Interaction Dataset* 4

[4] Kiwon Yun, Jean Honorio, Debaleena Chattopadhyay, Tamara L. Berg, and Dimitris Samaras, *Two-person Interaction Detection Using Body-Pose Features and Multiple Instance Learning*. CVPRW, 2012. 4

[5] John C. Platt, *Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods*. ADVANCES IN LARGE MARGIN CLASSIFIERS, MIT Press, 1999. 5

[6] Learning structural SVMs with latent variables, Chun-Nam John Yu and Thorsten Joachims, ICML'09. 3