Scale Invariant Human Action Detection from Depth Cameras using Class Templates

Kartik Gupta and Arnav Bhavsar
School of Computing and Electrical Engineering
Indian Institute of Technology Mandi, India
kartik_gupta@students.iitmandi.ac.in, arnav@iitmandi.ac.in

Abstract

We consider the problem of detecting and localizing a human action from continuous action video from depth cameras. We believe that this problem is more challenging than the problem of traditional action recognition as we do not have the information about the starting and ending frames of an action class. Another challenge which makes the problem difficult, is the latency in detection of actions. In this paper, we introduce a greedy approach to detect the action class, invariant of their temporal scale in the testing sequences using class templates and basic skeleton based feature representation from the depth stream data generated using Microsoft Kinect. We evaluate the proposed method on the standard G3D and UTKinect-Action datasets consisting of five and ten actions, respectively. Our results demonstrate that the proposed approach performs well for action detection and recognition under different temporal scales, and is able to outperform the state of the art methods at low latency.

1. Introduction

Human action detection still remains as a very challenging task in computer vision. Action detection requires the localization of a particular action in the temporally unsegmented sequences consisting of multiple actions instances as depicted in Fig. 1. We believe that this is one reason which makes this task more challenging than action recognition on temporally segmented sequences. The ability to detect and temporally localize the human actions with low latencies is of utmost importance to several applications of action recognition such as video surveillance, video retrieval, sign language detection, automated driver assistance, and human-robot interactions. These applications call for a robust detection algorithm which can work on even complex actions in real time.

The problem of action detection involves considering the intra-class variations among different classes, viewpoint variation, and variation in temporal scale of the action sequence and the latency in detection of the action class. Intra-class variations occur due to the differences in the way the same action is performed by different subjects. Temporal scale of the action can drastically vary for the same action and a fixed scale search strategy to detect the action in a continuous sequence might not work well. Detection Latency can either be observational latency or computational latency as mentioned in [1], where observational latency is the time required by the system to observe enough frames to make a decision, whereas computational latency is the time required to perform the actual computation on a frame. Hence, the detection algorithm must be robust against variations, and efficient enough to perform well at a lower detection latency.

The feature representation required to represent a sequence or frames of the sequences must be invariant to translation, and illumination conditions. With the recent development in low cost depth sensors, such as Microsoft Kinect, it is now possible to perform pose estimation and capture the 3D skeleton data in realtime [12], invariant to variation in color and illumination conditions.

Previous works [1], [17] & [11] either use a fixed temporal scale approach for the training and testing of the action class or they use a multi scale approach. It is evident that a fixed scale approach is not the best strategy for action detection, as the same action can be performed for different time durations. The reason for using fixed scale approach in the previous works is the use of video level features which represent a sequence with a single feature vector instead of frame level features. The frame level feature representation helps in better addressing the temporal scale variance. We represent skeleton features at frame level after preprocessing as done in [14], which also achieve viewpoint invariance and spatial scale invariance of the skeletons.

An important aspect in the action detection problem is to deal with the unknown starting and ending locations of action frames. In this paper, we introduce a greedy approach

Figure 1. Illustration of action detection problem on G3D dataset searching for action - left hand punch.

to detect specific actions in a continuous sequence, addressing the problem of unknown starting and ending frames of the action class. We demonstrate that our action detection approach is scale invariant and also achieves a low detection latency.

Our approach involves the notion of class templates and frame to metaframe distance from [7]. However, the approach proposed in [7] has only been applied for an offline continuous action recognition on RGB data using a dynamic programming based approach. We build upon this using our greedy strategy for online action detection, and also use more robust skeleton based features for feature representation. Fig. 2 depicts the overall framework of action detection approach that has been proposed in this paper, where class templates are used to represent the specific action class and later used for the testing purpose. The class template based framework used for training and testing addresses the problem of large intra class variations, and also makes the testing framework robust enough to work with a low number of training examples. With more traditional template based models like DTW, HMMs and CRF, it is difficult to mitigate the above mentioned issues.

Our results demonstrate that the proposed approach clearly outperforms the existing methods for the action detection tests performed on G3D - gaming dataset [2] for fighting actions. While our work is primarily proposed for action detection, we also evaluate our approach for action recognition on the UTKinect-Action dataset [16], consisting of common actions with challenges of viewpoint variance, and inter-class and intra-class variations.

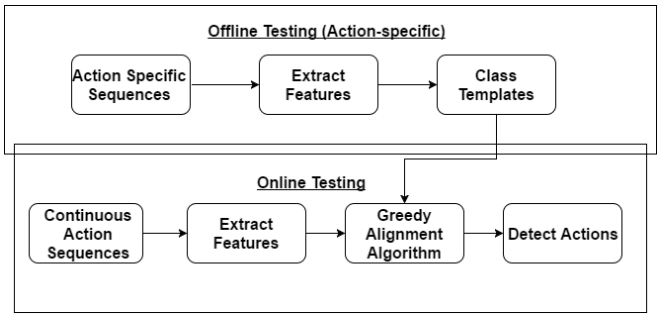

Figure 2. Overall framework of our action detection approach.

2. Related work

With the advent of real-time depth cameras, there has been much work on the problem of human action recognition rather than action detection. Vemulapalli *et al*. [14], Wang *et al*. [15], Oreifej & Lui [10] and Chen *et al*. [3] introduced robust feature representation techniques for problem of action recognition which perform effectively on the benchmark datasets. But these approaches use video level features to classify presegmented videos and haven't been extended to tackle the problem of action detection yet. Also, these work either will require fixed scale approach or a multi scale approach with high computational latencies, if directly used for action detection.

To measure latency, Nowozin and Shotton [9] introduced the concept of action points, a temporal anchor for action instances within a sequence. An action point is a single pose that can be clearly and easily identified as a representative of an action. Action points allow a very accurate and fine-grained detection for the actions. In Fothergill *et al*. [6], a fixed sliding window approach of 35 frames was used for

performing the action detection task.

Bloom *et al*. [1] proposed a dynamic feature selection based approach for online action recognition with a fixed scale strategy. They use multiple AdaBoost classifiers to train the reduced feature vectors using feature selection method built in to Random Forest. A fixed scale approach does not necessarily work for generic real-time sequence as the same actions can be performed in different durations.

Zhao *et al*. [17] detected and optimized the size of segment by dynamically matching with pre-learned templates. However, as the average length of their templates is about 35 frames, the observational latency of classification is high.

Sharaf *et al*. [11] proposed real-time multi-scale action detection using a descriptor derived from angles and angular velocities of the 3D joint data extracted from depth sensors. With multi scale approach, they are able to deal with scale variance to some extent but as the number of scales to detect are increased, there is a increase in computational latency as well. So, there is a tradeoff between the selection of scales and the computational latency.

Considering the limitations of related approaches mentioned above, we propose a greedy approach to detect actions in continuous action sequences, using class templates constructed from the training data. We demonstrate that our approach yields a good detection performance even under different temporal scales of actions, and also outperform related contemporary approaches in real time. Thus, the advantage of using our greedy alignment approach is its ability to handle temporal scale variance while working at a low latency for action detection.

The rest of the paper is organized into the following sections. In Section 3, we discuss the proposed approach in detail, outlining the Skeleton feature representation and the Detection framework. We present the experimental results and comparisons, with related discussions for various cases in section 4. We conclude in section 5.

3. Proposed approach

Human Action Detection from depth cameras requires 3D video representation (extraction of suitable spatio-temporal features) and machine modeling of human actions (modeling and learning of dynamic patterns) for training of the action classes and suitable pattern searching framework for 3D action detection for testing.

3.1. 3D video representation

We use 20 skeletal coordinates for each frame to represent that frame by processing the skeletal data to handle viewpoint variance, bone length scale variance and location variance. We make the skeletal data invariant to absolute location of the human in the scene, by transforming all 3D joint coordinates from the world coordinate system to a person-centric coordinate system by placing the hip center at the origin.

As mentioned in [14], we take one of the skeletons as reference, and normalize all the other skeletons (without changing their joint angles) such that their body part lengths are equal to the corresponding lengths of the reference skeleton. This normalization makes the skeletons bone length scale-invariant.

We also rotate the skeletons such that the ground plane projection of the vector from left hip to right hip is parallel to the global x-axis. This rotation makes the skeletons view-invariant.

Finally, we concatenate all the 20 joints x, y and z coordinates to represent each frame using 60 dimensional feature vector. We get feature vector f_i to represent i^{th} frame as:

$$f_i = [x_{i1}, y_{i1}, z_{i1}, x_{i2}, y_{i2}, z_{i2}, ..., x_{i20}, y_{i20}, z_{i20}]^T \quad (1)$$

3.2. Action detection framework

Our overall Action detection framework consists of two phases: Training phase, which is an offline phase where we represent each action class consisting of metaframes using notion of class templates as done in [7] and Testing phase, which is an online phase where we use a greedy algorithm with sliding initial frame search approach to detect action class from the continuous sequence. We note that OP-DFW and TP-DFW methods proposed in Kulkarni *et al*. [7] only allow offline recognition on continuous video sequences whereas our greedy approach targets to work on the problem of real time action detection.

3.2.1 Training phase

In [7], Kulkarni *et al*., presented a concept of action template, which is denoted by Y^l for each action category l and class template denoted by $\tilde{Y}^l$. For each class, among the training sequences $\{X_n^l\}$ we seek the sequence $Y^l \in \{X_n^l\}_{n=1}^{N_l}$, which represents the mean action of a class l. This can be done via the following minimization using dynamic time warping [8]:

$$i^* = \underset{i}{\operatorname{argmin}} \sum_{j \neq i} \text{DTW } \left(X_i^l, X_j^l\right) \quad (2)$$

where DTW is matching score between two temporal sequence using dynamic time warping.

The above equation aligns each example X_j^l with Y^l and, eventually, the class template can be defined as:

$$\tilde{Y}^l = \left(\tilde{y}_1^l, \ldots \tilde{y}_{t'}^l, \ldots, \tilde{y}_{T_{Y^l}}^l\right), \quad (3)$$

composed of a sequence of metaframes, where each metaframe $\tilde{y}_{t'}^l$ being a collection of matched frames resulting from the minimization (4).

$$\tilde{y}_{t'}^l = \left\{x_{jt'}^l\right\}_{j=1}^{j=N_{t'}^l}, \quad (4)$$

where $x^l_{jt'} \in X^l_j$ is associated with the t'-th metaframe, $x^l_{jt'} \in X^l_j$, and $N^l_{t'}$ is the number of frames associated with this metaframe.

3.2.2 Testing phase

The testing phase requires a measure for calculating frame to metaframe distance and a detection framework which can accomodate lack of knowledge about the boundaries of the action in the continuous test sequence.

Frame to metaframe distance To align a test sequence consisting of frames and class template consisting of metaframes, we require to find distance between the test frame z_t and metaframe $\tilde{y}^l_{t'}$. The frame-to-frame distance $d(x_t, y_{t'})$ can be used by the dynamic time warping algorithm but not with the framework dealing with class templates.

In order to compute a frame-to-metaframe distance, a test frame $z_t \in \mathbb{R}^K$ is taken as a linear combination of the training frames associated with a metaframe $\mathbf{X}^l_{t'} = [x^l_{1t'} \dots x^l_{jt'} \dots x^l_{N^l_{t'}t'}]$ and because only a few training frames are likely to be similar to the test frame, a sparse solution for $w_{t'}$ by solving the following basis pursuit denoising problem Chen *et al*. [4], is found.

$$\overline{w}_{t'} = \underset{w}{\operatorname{argmin}} \|z_t - \mathbf{X}^l_{t'} w\|_2 + \gamma \|w\|_1. \tag{5}$$

The frame-to-metaframe distance in closed-form reproduced from [7] and proposed in Evangelidis and Psarakis [5] is:

$$\tilde{d}\left(z_t, \tilde{y}^l_{t'}\right) = \min_w \left\| z_t - \frac{\overline{\mathbf{X}}^l_{t'} w}{\|\overline{\mathbf{X}}^l_{t'} w\|_2} \right\|^2_2 \; s.t. \sum_{i=1}^{|\mathcal{S}|} w_i = 1. \tag{6}$$

Algorithm 1 Greedy alignment algorithm

Input: Test sequence: $X_{1\to T_X}$ & Action Class l Template: $\tilde{Y}^l_{1\to T_{\tilde{Y}^l}}$
for $\Lambda = 1$ to $T_X - T_{\tilde{Y}^l}$ **do**
  Set $i = \Lambda$ & $j = 1$ & $Score = 0$
  while $i < T_X - \varphi$ & $j < T_{\tilde{Y}^l} - \varphi$ **do**
    $i^*, j^* = \underset{i,j}{\operatorname{argmin}}\, \tilde{d}(X_{i\to i+\varphi-1}, \tilde{Y}^l_{j\to j+\varphi-1})$
    $Score = Score + \tilde{d}(X_{i^*}, \tilde{Y}^l_{j^*})$
    repeat and set $i = i^*$ and $j = j^*$
  end while
  if $Score < \tau^l$ **then** $\chi = X_{\Lambda\to i}$
  end if
end for

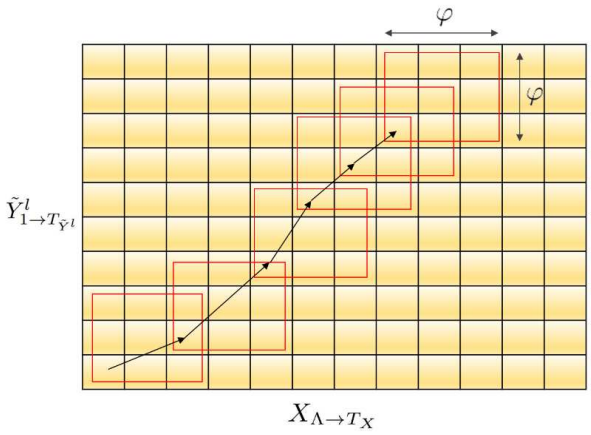

Figure 3. Illustration of Greedy alignment algorithm to match continuous test sequence $X_{\Lambda\to T_X}$ & class template $\tilde{Y}^l_{1\to T_{\tilde{Y}^l}}$ where within each local window of size φ, we search for the best match between frames and metaframes. Each grid point represents the frame to metaframe distance between respective metaframe of class template and the frame of test sequence.

Greedy alignment algorithm As mentioned earlier, for action detection it is imperative to locate the starting and the ending frames of an action class. we present a novel matching framework based on a greedy algorithm, which is depicted in Algorithm 1. For searching the starting frame, we propose a initial-frame search for class template $\tilde{Y}^l$ of length $T_{\tilde{Y}^l}$ on a continuous test sequence X of length T_X. The class template $\tilde{Y}^l$, during each iteration is slided on the test sequence X starting at Λ, to search for the beginning of the action class.

As we do not know the end point of the test action in the test sequence, a dynamic programming approach such as DTW or other template based models cannot work here. To locate the ending frame, and hence the overall matched segment, we search for best match between the metaframes of class template $\tilde{Y}^l$ and frames of continuous test sequence X within a local alignment window as illustrated in Fig. 3 using:

$$i^*, j^* = \underset{i,j}{\operatorname{argmin}}\, \tilde{d}(X_{i\to i+\varphi-1}, \tilde{Y}^l_{j\to j+\varphi-1}) \tag{7}$$

where i, j are the indices of the alignment window and i^*, j^* are the indices of the best local match in the alignment window and φ is size of the alignment window. This frame to metaframe distance for the best matches as computed above, is accumulated as the matching $Score$.

The size of the square alignment window which we use at each iteration of frame matching in the Algorithm 1 is calculated as :

$$\varphi = min(\lfloor T_X/10 \rfloor, \lfloor T_{\tilde{Y}^l}/10 \rfloor) \tag{8}$$

With the above approach, we get some predicted segments χ for the test action class in the continuous test sequence based on the matching threshold τ^l using the Algorithm 1. We set the threshold τ^l as the closest matching score we get, when training the action class template $\tilde{Y}^l$ with all the training examples of the action class l. Ideally, we should get a single predicted segment χ for each action class l, for which the score is lesser than the threshold. However, if multiple instances of predicted segments fall below the threshold, it would reflect in lowering of the F_1-score measure (based on precision and recall).

We importantly note that we train and detect each action independently of the other actions in the test sequence. That is, to detect a particular action, we do not require the approach to be trained on the other actions present in the test video.

Action recognition We also evaluate our framework for Action Recognition on the UTKinect-Action dataset for comparative analysis as there are no previous works for action detection on this dataset. Training phase for the problem remains the same as we deal with temporally presegmented sequences in both action detection and action recognition. The only difference that comes in greedy alignment algorithm is that we do not have to make initial frame search as the test sequences are already temporally presegmented in action recognition.

4. Experiments and results

We evaluate our approach on two standard datasets: G3D [2] and UTKinect-Action dataset [16]. We also show comparisons of our work with some state-of-the-art methods.

4.1. Datasets

The G3D dataset comprises of 10 subjects performing 20 gaming actions grouped into seven categories. We select fighting category for evaluation of proposed strategy as it has substantial temporal scale variance, inter class variations and intra class variations. To our knowledge, all previous works which use the G3D dataset for action detection, evaluate their approach only on the fighting category. The fighting category contains five gaming actions: right punch, left punch, right kick, left kick and defend.

UTKinect-Action dataset contains 10 types of human actions in indoor settings using a single stationary Kinect. The sampling rate for both RGB and depth images is 30 frames per second (FPS). Skeleton coordinates for the detected skeletons for each frame are available. The 10 actions include: walk, sit down, stand up, pick up, carry, throw, push, pull, wave and clap hands. Each action was collected from 10 different persons for 2 times: 9 males and 1 female. The reason for evaluating our approach on this dataset is that this dataset is more complex than G3D dataset considering the fact that it contains periodic actions and has substantial temporal scale variance and viewpoint variance.

Both the datasets consist of continuous action sequences with multiple actions being performed by several subjects.

4.2. Experimental settings

We evaluate proposed approach on the leave-persons out protocol as used in previous works to make comparisons more informative. We have used the same evaluation measures as in other approaches for comparative results. For the G3D dataset, all actors perform all the actions. We remove one subject from the full dataset to construct the test set and the larger remaining set of videos is used for training (i.e. 27 videos for training and 3 videos for testing in each fold). This process is repeated 10 times with different subjects to obtain the general performance.

For UTKinect-Action dataset we remove five subjects for training and use the removed set of 5 subjects for the testing purpose (i.e. 10 videos for training and 10 videos for testing for each fold). We perform the experiments two times taking at first odd numbered subjects as training purpose and then even numbered subjects for training purpose. Note that this strategy highlights the evaluation with lesser training examples.

4.3. Evaluation metrics

We need to use a latency-aware evaluation metric to test our action detection algorithm. To make the comparative analysis for G3D dataset, we use the action point annotations available in G3D dataset as used in previous works [1],[11]. Action point annotations are not available for UTKinect-Action dataset and there has been no previous work on the problem of action detection on this dataset. Also, it comprises of periodic actions where the motion is repetitive like walking, clapping, etc. Hence, for UTKinect-Action dataset we assign a detected action as true positive if more than 50% of the frames for both ground truth segment and detected temporal segment of the action match.

To find the latency in detection in UTKinect-Action dataset, middle frame index of the detected segment and middle frame index of the ground truth segment are assigned as predicted action point t_p and ground truth action point t_g respectively. The system latency shows how large the delay between the true action point and the system prediction is. For a specified amount of latency (Δ), a detection made at time t_p for action a is correct with respect to a ground truth action point at time t_g, if $|t_g - t_p| \leq \Delta$. For G3D dataset, we use ground truth action point annotation available for each sequence as t_g and time to match action point of the class template with the predicted segment χ, as t_p. We employ the F_1-score defined as:

$$F_1(a, \Delta) = 2\frac{precision_a(\Delta).recall_a(\Delta)}{precision_a(\Delta)+recall_a(\Delta)} \quad (9)$$

where $precision_a(\Delta)$ is the fraction of retrieved instances of action class a that are relevant, and $recall_a(\Delta)$ is the fraction of relevant instances of action class a that are retrieved.

Because the system detects multiple actions, we used the mean F_1-score over all actions:

$$F_1(A, \Delta) = \frac{1}{|A|} \sum_{a \epsilon A} F_1(a, \Delta) \quad (10)$$

where A is the set that contains all the different actions in the dataset.

4.4. Action detection results

We perform detection experiments on G3D dataset and UTKinect-Action dataset. Computational latency is very low for our proposed detection framework as our method runs in real time ($\sim$100 fps). The run-time is obtained on a notebook with a 2.10 GHz core i3 CPU and 6 GB RAM. The implementation is using an unoptimized and unparallelized MATLAB code.

Actions	F_1-Score	(Min, Max) frames	Mean no. of frames	Std. dev. of no. of frames
Right Punch	1	(4, 35)	13.63	6.47
Left Punch	0.933	(6, 26)	11.57	5.33
Right Kick	0.933	(3, 25)	8.60	5.33
Left Kick	0.867	(3, 22)	8.73	4.89
Defend	0.967	(20, 101)	40.37	16.32
Average	0.94	(7.2, 41.8)	16.58	7.66

Table 1. Performance of our method on G3D dataset and temporal scale indication of each action of 'Fighting' category.

Table 1 shows the performance of our method for the G3D Fighting dataset, where we note that we achieve a very high F_1-Score, consistently for all categories. Table 2 shows the comparative results between our approach and the state of the art detection approaches on fighting category of G3D dataset. Clearly, our approach even while using with simple skeleton based features, outperforms all the existing approaches representation at low observational latency (Δ) = 333 ms.

Method	Avg. F_1-Score
Adaboost[1]	0.896
Dynamic feature selection[1]	0.919
Multi Scale Detection[11]	0.937
Ours	0.94

Table 2. Comparisons of our method with the state-of-the-art approaches using leave-one-out protocol on the 'Fighting' category of G3D dataset.

Table 3 shows the performance for the various actions of UTKinect-Action dataset and also the frame scale variance for each action category in UTKinect-Action dataset. Note that this dataset is somewhat more complex than the G3D dataset involving complex periodic actions, larger temporal scale variance, viewpoint variance. Still, at such complexity, our approach achieves a high F_1-score = 0.8177 on this dataset at observational latency (Δ) = 333 ms.

The low detection scores for the cases of *walk* and *carry* action are because these actions are also present in parts of some other actions, i.e. we have multiple instances of these actions in a single sequence, even though only one of them is labeled as *walk* or *carry*. Also, both *carry* and *walk* themselves involve the walking action. As the skeleton based features are very noisy in the case where subject bends down to pick up an object, it is not modeled well, hence action class *pick up* results in lowest detection score. Action class *throw* comprises of very similar motions by the subject to the action class *push* and *pull* combined, hence *throw* has a low precision, whereas *push* and *pull* independently are modeled with high detection scores. We also want to emphasize here that our feature representation is quite simple, and the detection rates may be further improved using more sophisticated feature representation.

The last three columns of Table 1 and Table 3 indicate the temporal variations in the number of frames for the actions in this dataset. These results highlight that our approach maintains its good performance even with large scale variations among actions. This scale invariant performance of our method can be noted across both the datasets.

4.5. Action recognition results on UTKinect-Action dataset

While we demonstrated encouraging results for action detection on the UTKinect-Action dataset, we did not provide comparative results as there are no previous works available on UTKinect-Action dataset for the problem of action detection. However, as discussed towards the end of section 3.2.2, the proposed approach can also be modified in a straight-forward manner for action recognition, for which there have been some works reported on the UTKinect-Action dataset. We believe that a positive comparative results on this dataset for action recognition would also reflect on a superior detection performance.

For action recognition, we presegment the continuous sequences as the respective action sequences based on the segment labels for each action present in the dataset, to use this dataset for action recognition problem. Table 4 shows the comparative results of some state of the art skeleton based methods of action recognition experiments performed on UTKinect-Action dataset for 5-5 cross subject protocol. Fig.4 shows more detailed recognition analysis for all the actions in UTKinect-Action dataset.

Actions	Precision	Recall	F_1-Score	(Min, Max) frames	Mean no. of frames	Std. dev. of no. of frames
Walk	0.5	0.6	0.545	(22, 54)	40.85	8.07
Sit Down	1	1	1	(15, 48)	33.25	9.01
Stand Up	1	1	1	(13, 48)	24.65	6.50
Pick Up	0.33	0.4	0.362	(14, 55)	35.20	11.40
Carry	0.727	0.8	0.762	(1, 110)	49.45	22.99
Throw	0.538	0.7	0.608	(5, 19)	12.15	4.02
Push	1	1	1	(6, 17)	9.85	3.08
Pull	1	1	1	(7, 29)	13.65	5.90
Wave Hands	1	1	1	(22, 70)	44.05	14.23
Clap Hands	0.9	0.9	0.9	(14, 72)	28.50	15.36
Average	0.7995	0.84	0.8177	(11.90, 51.20)	29.16	10.06

Table 3. Performance of our method on the UTKinect-Action dataset using 5-5 cross-subject evaluation scheme, and temporal scale indication of each action category.

Method	Accuracy
HOJD[16]	90.92%
Random Forests[18]	91.90%
Grassmannian representation[13]	95.25%
Ours	96%

Table 4. Recognition Accuracy comparisons of our method to some state-of-the-art skeleton based approaches using 5-5 cross subject evaluation of UTKinect-Action dataset.

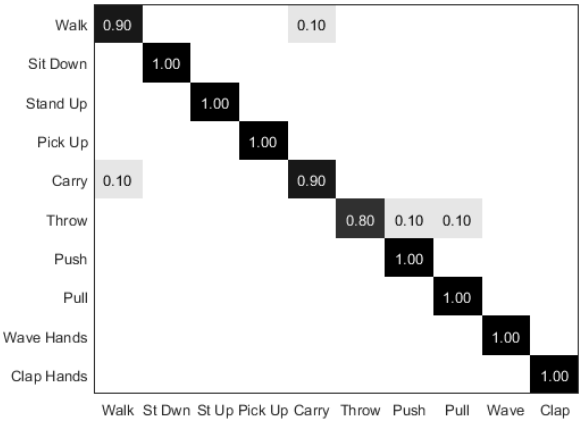

Figure 4. Recognition accuracy of UTKinect-Action dataset consisting of 10 actions.

The recognition results clearly show that our approach shows positive comparisons with respect to some contemporary methods. Also, we note that the training data in this case is quite low, with just 5 subjects for training for each class, and rest of the 5 for the testing. The possible reason behind the efficacy of our approach is the use of frame level features and the scale invariance. For some of the other methods, periodic actions like *walk*, *carry*, *wave hands* and *clap hands* may tend to get misclassified due to variance in their scales for the same action.

It is also clearly evident that action classes *walk*, *carry*, *pick up* and *throw* have much better recognition accuracy in action recognition than detection rate in action detection. As the action recognition problem considers presegmented action sequences, we do not face the problem of multiple action instances of same action in the sequence and also the class templates are modeled very robustly.

Thus, we believe that frame level feature representation, combined with the notion of class templates, and the resultant scale invariance and performance at low latency, are important features of approach.

5. Conclusion

In this paper, we propose a novel scale invariant approach for the 3D human action detection, based on the notion of class templates, which is more robust to inter-class and intra-class variations. The framework employs simple skeleton joint based feature representation which has ability to capture both structural and spatio-temporal information. Our experiments and results show that the proposed method outperforms the existing 3D action detection methods on benchmark dataset of G3D dataset for fighting category. Our experiments also show that the proposed framework works well with lesser training examples on the complex UTKinect-Action dataset involving high viewpoint variance and temporal scale variance. Another overall advantage to this framework is that the observational and computational latency of the method is quite low which makes it an ideal choice for real time applications such as video retrieval or action localization.

References

[1] V. Bloom, V. Argyriou, and D. Makris. Dynamic feature selection for online action recognition. In *Human Behavior Understanding*, pages 64–76. Springer, 2013.

[2] V. Bloom, D. Makris, and V. Argyriou. G3d: A gaming action dataset and real time action recognition evaluation framework. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 7–12. IEEE, 2012.

[3] C. Chen, K. Liu, and N. Kehtarnavaz. Real-time human action recognition based on depth motion maps. *Journal of Real-Time Image Processing*, pages 1–9, 2013.

[4] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[5] G. D. Evangelidis and E. Z. Psarakis. Parametric image alignment using enhanced correlation coefficient maximization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(10):1858–1865, 2008.

[6] S. Fothergill, H. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1737–1746. ACM, 2012.

[7] K. Kulkarni, G. Evangelidis, J. Cech, and R. Horaud. Continuous action recognition based on sequence alignment. *International Journal of Computer Vision*, 112(1):90–114, 2015.

[8] M. Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[9] S. Nowozin and J. Shotton. Action points: A representation for low-latency online human action recognition. *Microsoft Research Cambridge, Tech. Rep. MSR-TR-2012-68*, 2012.

[10] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2013.

[11] A. Sharaf, M. Torki, M. E. Hussein, and M. El-Saban. Real-time multi-scale action detection from 3d skeleton data. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 998–1005. IEEE, 2015.

[12] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[13] R. Slama, H. Wannous, and M. Daoudi. Grassmannian representation of motion depth for 3d human gesture and action recognition. In *22nd International Conference On Pattern Recognition*, pages 3499–3504, 2014.

[14] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–595, 2014.

[15] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1290–1297. IEEE, 2012.

[16] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 20–27. IEEE, 2012.

[17] X. Zhao, X. Li, C. Pang, X. Zhu, and Q. Z. Sheng. Online human gesture recognition from motion data streams. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 23–32. ACM, 2013.

[18] Y. Zhu, W. Chen, and G. Guo. Fusing spatiotemporal features and joints for 3d action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 486–491, 2013.