Human Activities Recognition using Depth Images

Raj Gupta[1] Alex Yong-Sang Chia[2*] Deepu Rajan[1]

[1]School of Computer Engineering, Nanyang Technological University - Singapore
[2]Rakuten Inc. - Japan
{rajk0005, asdrajan}@ntu.edu.sg, alex.chia@mail.rakuten.com

ABSTRACT

We present a new method to classify human activities by leveraging on the cues available from depth images alone. Towards this end, we propose a descriptor which couples depth and spatial information of the segmented body to describe a human pose. Unique poses (i.e. codewords) are then identified by a spatial-based clustering step. Given a video sequence of depth images, we segment humans from the depth images and represent these segmented bodies as a sequence of codewords. We exploit unique poses of an activity and the temporal ordering of these poses to learn subsequences of codewords which are strongly discriminative for the activity. Each discriminative subsequence acts as a classifier and we learn a boosted ensemble of discriminative subsequences to assign a confidence score for the activity label of the test sequence. Unlike existing methods which demand accurate tracking of 3D joint locations or couple depth with color image information as recognition cues, our method requires only the segmentation masks from depth images to recognize an activity. Experimental results on the publicly available Human Activity Dataset (which comprises 12 challenging activities) demonstrate the validity of our method, where we attain a precision/recall of 78.1%/75.4% when the person was not seen before in the training set, and 94.6%/93.1% when the person was seen before.

Categories and Subject Descriptors

I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis

Keywords

Human activity detection; depth image segmentation

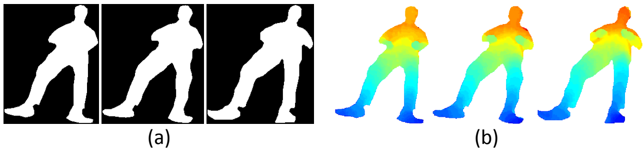


Figure 1: Silhouettes ambiguities in poses. Pose differences (due to different hand positions) cannot be seen from silhouettes in (a), but are readily visible from three-dimensional shape information in (b).

1. INTRODUCTION

Human activities recognition is useful in many applications like surveillance, action/event centric video retrieval and patient monitoring systems. A large majority of activity recognition works focuses on gray or RGB video sequences, and exploits spatio-temporal interest points or silhouettes as recognition cues. Here, a significant challenge is to sieve out image/video features that are representative of the activity in which background clutter, illumination changes and camera motion could easily corrupt these features. Also, given that these features incorporate only spatial x-y and temporal information, they often fail to fully resolve the silhouette ambiguity in poses under self occlusion [27].

There is an emerging consensus that depth information is a more reliable recognition cue for classifying human activities. It is color/texture/intensity invariant and hence is robust towards appearance variations of the humans performing the action. More importantly, recent depth cameras (e.g. Kinect) offer substantial depth resolutions of a few centimeters and thus provide very good estimation of the three-dimensional geometry of the scene. Unlike gray/color images acquired with traditional 2D cameras which provides spatial x-y image information, depth images afford another important spatial dimension (z-component) which can be exploited to help resolve silhouettes ambiguities in poses. For example, consider Fig. 1(a) which shows silhouettes of a person lying on the ground and moving his arms. Due to self-occlusion, the difference in poses is not evident. However, in Fig. 1(b), which shows the corresponding depth images, the outline of the hands is visible facilitating pose description.

In this paper, we present a method which harnesses *only* depth images to recognize very complex human activities such as brushing teeth, cooking - chopping and cooking - stirring. Our method does not demand knowledge of body joints locations, but only requires segmentation mask of the human silhouette to be available. We propose an algorithm

[*] Part of this work was done while Alex Yong-Sang Chia was with Institute for Infocomm Research - Singapore.

MM '13, October 21 - 25 2013, Barcelona, Spain
Copyright 2013 ACM 978-1-4503-2404-5/13/10 ...$15.00.

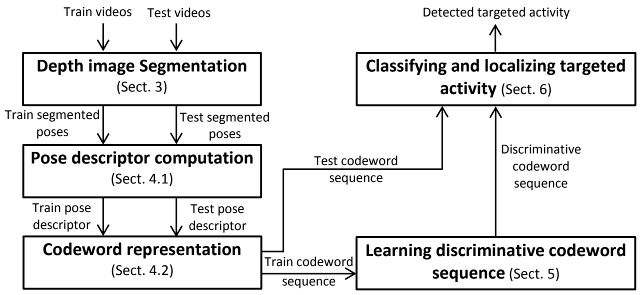

Train videos
Test videos
Depth image Segmentation (Sect. 3)
Train segmented poses
Test segmented poses
Pose descriptor computation (Sect. 4.1)
Train pose descriptor
Test pose descriptor
Codeword representation (Sect. 4.2)
Train codeword sequence
Learning discriminative codeword sequence (Sect. 5)
Test codeword sequence
Discriminative codeword sequence
Classifying and localizing targeted activity (Sect. 6)
Detected targeted activity

Figure 2: Block diagram of the proposed algorithm.

to extract the human silhouette from depth images. A scale and depth invariant descriptor is developed to represent the 3D pose of a segmented silhouette. We extract the descriptors from training images depicting a wide array of poses, and cluster these descriptors to find a set of compact clusters. Here, we propose a spatial based clustering step to ensure cluster members are well represented by its medoids, and consider each medoid as a codeword. Collectively, these codewords represent the range of unique body poses that are exhibited by the humans in the training images.

Fig. 2 shows the block diagram of the proposed algorithm. For a given training depth video sequence (and its activity label), we segment the human from the video sequence, and represent the sequence of segmented bodies as a sequence of codewords. Discriminative subsequence of codewords that model unique poses and the distinguishing temporal ordering of these poses are then learned with a boosting framework. Each discriminative subsequence has a variable number of codewords, and is termed in this paper as *x-subsequence-codewords* or *xSC*. Both the *xSC*, its window size and weight are learned automatically, and represent the most discriminative subsequences for the activity class. At test time, we adopt a sliding window approach to align each *xSC* to overlapping windows of the test sequence. Each *xSC* acts as a classifier and we learn a boosted ensemble of *xSC* to give a confidence score for the activity label of the test sequence. We evaluated our method on the Human Activity Detection database [30] which comprises 12 challenging activities, and obtain state-of-the-art recognition results. Additionally, we also applied our method on the task of activity detection where we localize all instances of a targeted activity from a test video sequence. Accurate localizations of targeted activities are demonstrated.

2. RELATED WORK

Majority of human activity recognition works [2] exploit gray/color cues, with the use of depth cues attracting interest only in recent years. Here, we first discuss the more representative recognition works which exploit gray/color videos, before discussing those that exploit depth images.

Human activity recognition works mainly employ either a holistic representation or a part-based representation. In holistic representation methods, Bobick *et al.* [6] developed the motion energy and motion history images to encode short spans of motions efficiently. Hu moments [12] extracted from these images are then exploited as activity descriptors. Efros *et al.* [9] exploited optical flow measurements to compute spatio-temporal descriptors, and demonstrated the power of these descriptors in recognizing activities seen in low resolution videos. Yilmaz *et al.* [32] tracked moving 2D contours to generate a 3D spatio-temporal volume, and extracted differential geometry features from the surfaces of the volumes to recognize various activities. Although their method achieved very good activity recognition accuracy, it demands robust tracking of the contours and fails when contours cannot be tracked (e.g. under self-occlusion). Parameswaran *et al.* [23] used 3D joint locations to model the geometry of an action, and represented an activity in terms of its static canonical poses and dynamic trajectories in 2D invariance space. A limitation of this work is the need for accurate 3D joint locations.

Motivated by the success in the object classification domain, part based representations for activity recognition have gained strong traction in the research community. The majority of these methods employ sparse interest point descriptors such as [15, 8, 20] to recognize human actions. For example, Schuldt *et al.* [26] represented each video sequence by a bag-of-word representation, in which they employed STIP features [15] as the underlying features. A support vector machine is then learned for classifying test video sequences. Very good results have been demonstrated on videos depicting simple actions like running and walking. Dollar *et al.* [8] applied separate linear filters in the spatial and temporal dimensions to detect interest points that have local maxima value in both dimensions. A classifier is then learned in a similar way as [26] to recognize a test activity. The above methods use a histogram representation of features and ignore the temporal ordering between features. To preserve the temporal ordering, Nowozin *et al.* [20] improved on the histogram representation and proposed a sequential representation to preserve the temporal ordering between words. The PrefixSpan subsequence mining algorithm is exploited to find discriminative representation of an activity. Their method does not model the global geometry of human body joints when performing an activity, and instead considers them as a simple bag of features.

The above methods exploit appearance (color/texture) information as the key recognition cues and are often easily corrupted by variations in illumination and appearance of the human body. More importantly, such features encode the spatial (x-y) information, and cannot fully resolve silhouette ambiguity seen in self-occlusion [27]. To overcome these problems, recent works use depth cues to recognize human activities. Ni *et al.* [19] and Zhao *et al.* [34] used multi-modality sensor combination (e.g. color and depth) to compute Depth-Layered Multi-Channel STIPs and three dimensional motion history images for human activity recognition. Hao and Parker [33] extended Dollar's method [8] using depth images and constructed a 4D hyper cuboid for feature extraction. While these methods demonstrated improved performance gained by these depth extended feature representations, the use of color images affect the performance of these algorithms due to the reasons discussed earlier. Li *et al.* [17] proposed the first work that uses only depth information to recognize human activities. Here, they sampled a bag of 3D points from depth images, and project these points into a 2D space. An action graph is constructed from the training points to encode the actions to be recognized. Although their method achieves good recognition accuracy, the use of 2D projections of key poses can lead to sub-optimal feature representations. To describe the pose of

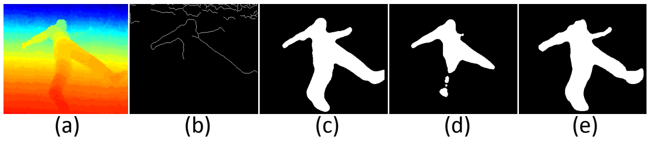


Figure 3: Depth image segmentation. (a) Depth image where an object is very near to the background. (b) Edge map extracted from (a). (c)-(d) Segmentation masks obtained by applying background subtraction [35] on color image and depth image respectively. (e) Mask obtained with our method which is able to extract the object completely (similar to (c)) with depth image alone.

human, Shotton *et al*. [27] exploited decision trees and artificially rendered humans to estimate 3D joint locations from a single depth image. This method was further extended by Holzer *et al*. [11] in which regression trees were used to learn interest points from depth images. Excellent joins/interest point detection were shown in [27, 11], though application of detected joints/points to human activity recognition tasks were not demonstrated. Sung *et al*. [29, 30] directly used the orientations and 3D joint locations to learn a model based on maximum-entropy Markov model. To deal with inaccurate joint locations, Wang *et al*. [31] used *local occupancy pattern* around each joint location detected by the tracker. While [29, 30, 31] yield very good classification of complex activities, it require the detection of body joint locations. This limits their applicability.

3. DEPTH IMAGE SEGMENTATION

The segmentation of moving objects from a depth image is an important problem. While standard segmentation approaches [18, 35, 28] that are developed for RGB images can also be used for depth images, these approaches often fail to extract objects that are very close to the background (e.g. person lying on the ground/bed, objects very near to wall) and require additional information like color for proper segmentation. Fig. 3 shows such an example where the person is lying on the ground. Since the object is very near to the background, the object boundaries become ambiguous in the depth map. This is illustrated in Fig. 3(b), which shows the edge map of Fig. 3(a) obtained by the Canny edge detector. Fig. 3(c) and 3(d) show the segmentation results obtained after applying a popular background subtraction method [35] on color and depth images, respectively, where segmentation masks obtained from depth image is noticeably weaker. To address this issue, we propose a method that computes the spatiograms [5] of the depth image along with the projections of its surface normals in xy, yz and zx planes for both images (background image and object+background image) and then compares these spatiograms to compute the similarity scores [21]. These similarity scores are used to segment the foreground object as shown in Fig. 3(e).

3.1 Extraction of surface normals

We briefly review the method to extract surface normals as described in [25]. Let the position vectors of points in an image be given as a function of $(s,\ t)$ coordinates of the parametric space (image coordinates). We define $\vec{X_s}$ and $\vec{X_t}$ as the partial derivatives of $\vec{X}$ with respect to s and t,

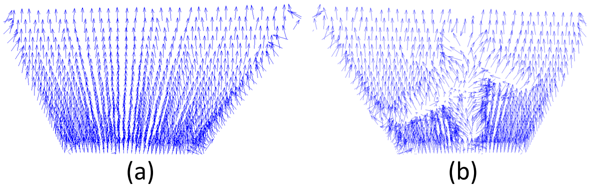


Figure 4: Visualization of surface normals computed on (a) background depth image and (b) the object+background depth image (shown in Fig. 3(a)).

$$\vec{X} = \begin{pmatrix} x(s,\ t) \\ y(s,\ t) \\ z(s,\ t) \end{pmatrix}; \vec{X_s} = \begin{pmatrix} x_s(s,\ t) \\ y_s(s,\ t) \\ z_s(s,\ t) \end{pmatrix}; \vec{X_t} = \begin{pmatrix} x_t(s,\ t) \\ y_t(s,\ t) \\ z_t(s,\ t) \end{pmatrix}. \quad (1)$$

For each of the functions $x(s,\ t)$, $y(s,\ t)$, and $z(s,\ t)$ a local least squares polynomial function fit is evaluated. This polynomial function is then differentiated to compute the derivatives $\vec{X_s}$ and $\vec{X_t}$. Given a function $f(s,\ t)$, the derivatives are evaluated as [4]: $f_s = D_s * S * f$ and $f_t = D_t * S * f$ where $*$ denotes convolution and S is a smoothening operator. D_s and D_t are given respectively as $\vec{C_0}\ \vec{C_1}^T$ and $\vec{C_1}\ \vec{C_0}^T$, where $\vec{C_0} = \frac{1}{7}\ [1\ 1\ 1\ 1\ 1\ 1\ 1]^T$ and $\vec{C_1} = \frac{1}{28}\ [-3\ -2\ -1\ 0\ 1\ 2\ 3]^T$. The surface normal is then computed as the cross product of the partial derivatives $\vec{X_s}$ and $\vec{X_t}$,

$$\hat{n} = \frac{\vec{X_s} \times \vec{X_t}}{\parallel \vec{X_s} \times \vec{X_t} \parallel}. \quad (2)$$

Fig. 4(a) shows the surface normals computed for a background image, while Fig. 4(b) shows the surface normals computed on Fig. 3(a), which contains both the foreground object and the background. For each of these surface normals, the projections onto the xy-plane, the yz-plane, and the zx-plane are used to compute the spatiograms [5].

3.2 Spatiogram computation and matching

A spatiogram [5] is a generalization of histogram in which feature distribution information of a histogram is combined with spatial layout information (mean and covariance of the spatial position of all pixels that fall into each bin). This allows spatiograms to capture higher-order spatial moments of each attribute bin. To compute the histogram for an image of N pixels, the histogram bin count n_b of bin b can be written as:

$$n_b = C \sum_{i=1}^{N} \delta_{ib}, \quad (3)$$

where C is a normalizing constant and

$$\delta_{ib} = \begin{cases} 1 & \text{if } i^{th} \text{ pixel falls in } b^{th} \text{ bin} \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Let $X_i = [x_i, y_i]^T$ be the spatial position of pixel i, in which the spatial co-ordinates in the image are normalized to $[-1, +1]$. For each bin, we compute the spatial mean (μ_b) and covariance (Σ_b) as:

$$\mu_b = \frac{1}{\sum_{j=1}^{N} \delta_{jb}} \sum_{i=1}^{N} X_i \delta_{ib}, \quad (5)$$

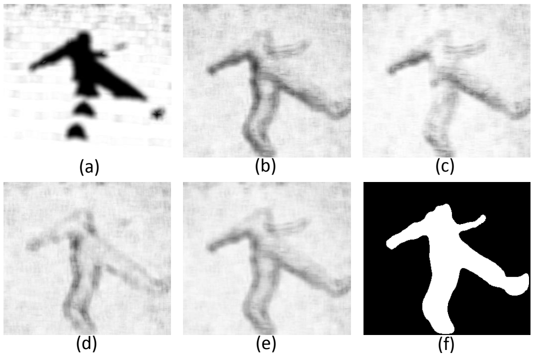


Figure 5: Illustration of similarity between spatiograms obtained for background only image and object+background image. The spatiograms are computed for (a) depth image, and projections of surface normals onto (b) xy, (c) yz and (d) zx planes. (e) Average similarity score of surface normal projections. (f) Segmentation mask obtained using similarity maps (a) and (e).

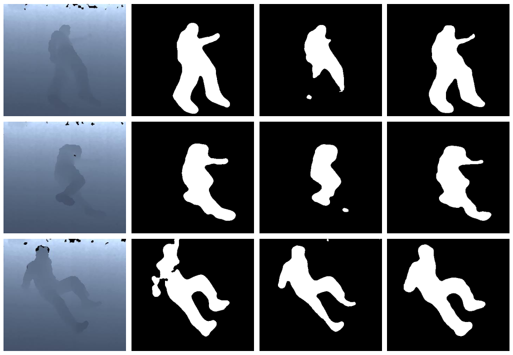


Figure 6: Results of depth image segmentation. First column: Input depth image. Second and third columns: Segmentation mask obtained after applying standard background subtraction method [35] on color and depth images respectively. Fourth column: Segmentation mask obtained by our algorithm.

$$\Sigma_b = \frac{1}{\sum_{j=1}^{N} \delta_{jb}} \sum_{i=1}^{N} (X_i - \mu_b)(X_i - \mu_b)^T \delta_{ib}, \qquad (6)$$

Given a background image and an object+background image, we compute spatiogram $S = \{n, \mu, \Sigma\}$ for their depth images and for the projections of their surface normals in xy, yz and zx planes. Let $S = \{n, \mu, \Sigma\}$ and $S' = \{n', \mu', \Sigma'\}$ denote respectively the spatiograms for the background image and object+background image respectively. The similarity (ρ) between the two spatiograms is then computed as [21]

$$\rho = \sum_{b=1}^{B} \sqrt{n_b n_b'} \left[8\pi |\Sigma_b \Sigma_b'|^{1/4} N(\mu_b; \mu_b', 2(\Sigma_b + \Sigma_b')) \right], \qquad (7)$$

where $N(X; \mu, \Sigma)$ represents a normalized Gaussian evaluated at X. The similarity is computed at every pixel between spatiograms obtained for background depth image and object+background depth image. Similarities are also computed for the spatiograms obtained from the projections of the surface normals onto the three planes with the similarity computed, as before, between background only and object+background cases. The similarity maps ρ_d, ρ_{xy}, ρ_{yz} and ρ_{zx} for the depth image and all three projections of its surface normals in xy, yz and zx planes are shown in Fig. 5(a)-(d) respectively. Darker pixel represents higher dissimilarity between a background only image and an object+background image. These similarity maps are then combined together to detect the object position as follows:

$$mask = \begin{cases} 1 & \min(\rho_d,\ \rho_{xyz}) \leq T \\ 0 & \text{otherwise.} \end{cases} \qquad (8)$$

where T is a threshold and the value of ρ_{xyz} is

$$\rho_{xyz} = (\rho_{xy} + \rho_{yz} + \rho_{zx}) \,/\, 3, \qquad (9)$$

and is illustrated in Fig. 5(e). The final segmentation mask, shown in Fig. 5(f), is obtained after morphological operations to fill holes and to smoothen object boundaries.

3.3 Segmentation results

In this section, we compare the performance of the proposed algorithm with a popular background subtraction method [35]. To compute the similarity score at pixel p, we compute the spatiograms over a neighborhood of p for background and object+background images and then find the similarity between these two spatiograms using Eq. (7). In our experiments, we fix the neighborhood size and number of bins to 15 × 15 and 24, respectively. The threshold value T (in Eq. 8) has been chosen empirically and was fixed to 0.86 in all our experiments.

Fig. 6 shows segmentation results obtained by our method and by [35]. Here, we use depth images as input for the proposed method while we apply [35] on both color and depth images to demonstrate the effectiveness of the proposed algorithm. Input depth images are shown in the first column of Fig. 6. We show segmentation masks obtained by applying [35] on color and depth images in the second and third columns of Fig. 6 respectively. The segmentation masks obtained by our method are shown in the fourth column of Fig. 6. It can be seen that the proposed method is able to extract the complete objects from the depth images that are very close to the background.

4. CODEBOOK OF HUMAN POSES

We build a codebook of body poses from training images, and represent a human pose shown in a video frame by its most similar codeword. Here, explicit effort is made to learn codewords which not only have coherent appearances with a large number of poses (i.e. can represent most poses well), but also have substantial different appearances from other codewords (i.e. effectively covers the range of poses seen in the training set). To learn this codebook, we segment the human body from training depth images, and represent each segmented silhouette by a scale, depth and translation invariant descriptor. We apply a spatial-based clustering method to these descriptors, and consider the medoid of each resulting clusters as a codeword.

4.1 Computing pose descriptors

We require a framework to compare poses. A naive method is to consider a segmented depth image to be a 3D point cloud and compare two poses by the alignment of their 3D point clouds [24]. The alignment of 3D point clouds is a registration problem, which has high computational demands.

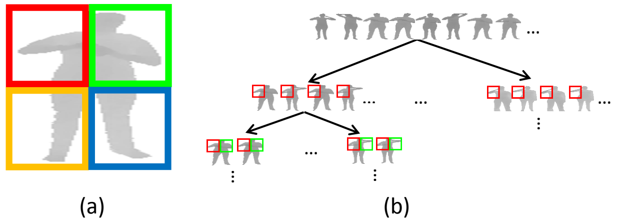


Figure 7: Spatial-based clustering. (a) Division of bounding box of a segmented body into 4 quadrants. (b) Clustering proceeds in the form of a hierarchical tree, in which we apply clustering on descriptor values of cells belonging to a quadrant at each tree level. Note that members in each sub-cluster have similar sub-poses within the processed quadrants (shown color coded in (b)).

For better efficiency, we extract descriptors from training images depicting a wide array of body poses, and exploit these descriptors to compare between poses. To ensure a descriptor is translation and scale invariant, we first segment the human from the background using the method presented in Sect. 3, and resize the segmented body to a canonical size prior to computing the descriptor. To describe the segmented body, a straightforward option would be a histogram of normalized depth values computed over the body, which is a simple bag of features representation. However, we can empower the descriptor with greater discriminative potential by encoding the spatial layout of the normalized depth values. Here, we define a dense grid over the bounding box of the segmented body, and compute the average normalized depth value for each cell in the grid. Additionally, we also compute the fraction of the body that is within each cell in the grid. The former incorporates depth information into the descriptor, while the latter encodes the spatial layout of the pose. The concatenation of these two values across all cells yields the pose descriptor.

4.2 Spatial-based clustering to learn codewords

We cluster the set of descriptors that are extracted from the training set, and compute the medoid of each cluster as a codeword. Here, we seek codewords which are sufficiently flexible such that each codeword can well represent poses that have similar appearances, but yet are substantially distinct from each other to ensure they cover the possible variations of poses. One approach to clustering is to consider each descriptor in its entirety during the the clustering step. However, such an approach considers every dimension in the descriptor *simultaneously*. Consequently, sharp differences between two descriptors in a few dimensions may be mitigated by small differences in other dimensions, so that dissimilar poses are grouped together into the same cluster.

To improve the quality of the clustering, we propose a spatial-based clustering step which explicitly incorporates pose information during clustering. We divide the bounding box of the segmented body into four quadrants (see Fig. 7), and cluster the descriptors based on their values within each quadrant separately. Here, we first apply clustering on the descriptor values of cells in the first quadrant to find sub-clusters whose members have similar poses within the first quadrant of the segmented body. The clustering step continues in the form of a hierarchical tree, in which each subsequent node of the tree corresponds to a sub-cluster formed from the previous clustering step, and we cluster on descriptor values from cells of quadrant i at level i of the tree. Consequently, each leaf node of the tree includes descriptors which have similar values in every quadrant, and hence are more likely to have similar poses. In our work, we apply the robust mean-shift clustering to the set of descriptors. Given that we apply mean-shift clustering on separate dimensions of the descriptors (as opposed to the entire dimensions collectively), and each application of mean-shift at a node of the tree is evaluated on sub-clusters with fewer members than its parent node, clusters can thus be efficiently found.

We apply the above method on the Human Action dataset [29, 30], and plot a histogram of the Euclidean distances between the cluster medoids (i.e. codewords) and its members (intra-cluster distances) in Fig. 8(a). For comparison, we show the histogram obtained by direct application of mean-shift clustering on the descriptor values in Fig. 8(b). Note that while both possess the same number of clusters, the proposed method finds much more compact clusters as shown by the smaller mean value (a t-test shows that this smaller mean value to be significant, $p < 10^{-15}$). Fig. 8(c) shows the histogram of the Euclidean distances between the cluster medoids found by the spatial-clustering method (inter-cluster distances), and Fig. 8(d) shows those obtained with direct application of mean-shift clustering. The proposed method finds codewords that are substantially distinct from each other, as shown by the higher mean Euclidean distance between codewords ($p < 10^{-12}$). We show codewords selected by our method in Fig. 9. As observed, salient and unique poses exhibited by the human, e.g. sitting, standing and drinking water are well represented.

5. DISCRIMINATIVE SUBSEQUENCES OF CODEWORDS

In this section, we describe our method to learn discriminative subsequences of video codewords (termed xSC in this paper) that are unique to a targeted activity. Inference at test time for the targeted activity is achieved by aligning these xSC to the test video codeword sequence with a sliding window approach, and computing the weighted sum of the confidence score at each window. Here, we first detail the learning of these activity-specific subsequence of codewords before discussing how they are exploited at test time.

We seek a discriminative xSC which models the distinguishing poses of a human when performing an activity, and the distinctive temporal ordering of these poses to reliably recognize an activity. The former incorporates pose/appearance features as recognition cues, while the latter encodes important temporal information that are unique to the activity. The alignment of these discriminative xSC to a window of a test sequence can thus be exploited to localize the activity within the test sequence with a sliding window approach. A key challenge here is on the *alignment* of a xSC to the test window. Specifically, a person may exhibit poses at test time that are slightly different from those seen during training, even when performing the same activity. Hence, the test video will be represented by a codeword sequence that is different from the xSC learned during training. More importantly, a person may perform a same activity with different speeds during training and testing. Such temporal variations are typically inconsistent across the activity (e.g. faster actions in initial phase of the activity, and slower

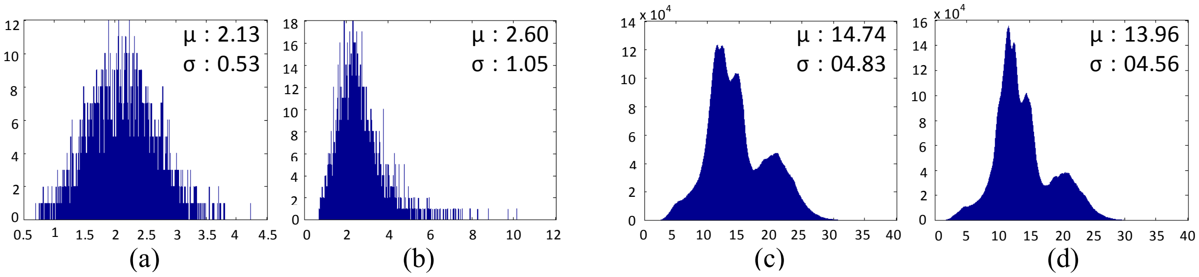


Figure 8: Comparison of clustering quality. Histogram of the Euclidean distances between codewords and its members obtained by our method and by direct application of mean-shift are shown in (a) and (b) respectively. Histograms of the Euclidean distances between codewords obtained by our method and by mean-shift are shown in (c) and (d) respectively. Notice that the proposed method finds more compact clusters, while learning codewords which have more diverse appearances. T-tests show the improvements to be statistically significant, ($p = 10^{-15}$ for comparison between (a) and (b), and $p = 10^{-12}$ for comparison between (c) and (d)).

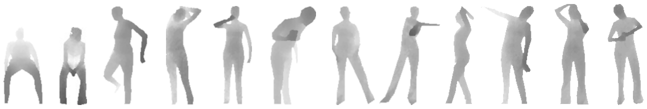


Figure 9: Codeword exemplars obtained for Human Activities dataset [29].

actions thereafter). This makes temporal ordering of the codewords in a xSC a potentially less reliable recognition cue. Here, we first describe our alignment method which addresses these two issues in a unified framework, before discussing how an ensemble of boosted xSC are learned.

5.1 Aligning xSC to a test window codewords

Motivated by recent successes of DNA sequences alignment in the bioinformatics field [14], we formulate the problem of video codewords alignment as a DNA sequence alignment problem where we consider a codeword of a video sequence to be analogous to a codon in a DNA sequence. Correspondingly, given a discriminative xSC and a test window of codewords, we compute the optimum alignment between the two sequences by the semi-global sequence alignment method [3]. Unlike the global alignment method, the semi-global alignment method ignores leading and trailing gaps in the sequence alignment process and is well suited for our problem in which a xSC may be longer than the test window codewords. Additionally, unlike local sequence alignment which finds *subsequences* of xSC within the test window, semi-global alignment searches for the presence of the *entire* xSC within the test window and hence fully exploits the discriminative potential of the xSC. In the following, we first describe the semi-global sequence alignment method, before addressing how this method gives robust alignment under pose and temporal variations in the video sequence.

Let P denotes a window of codewords, $P = \{p_1, \ldots, p_m\}$, where p_i is the i^{th} codeword in the sequence, and let $Q = \{q_1, q_2, \ldots, q_n\}$ denote a discriminative xSC. The optimum semi-global alignment between P and Q can be obtained by first computing a $(m+1) \times (n+1)$ score matrix M, where

$$M(i+1, j+1) = \min \begin{cases} M(i,j) + d(p_i, q_j) \\ M(i, j+1) + g \\ M(i+1, j) + g \end{cases} . \qquad (10)$$

Here, $d(p_i, q_j)$ denotes the Euclidean distance between the pose descriptors of codewords p_i and q_j, and g is a gap penalty which represents the cost of inserting a gap in the alignment. In all experiments, we set g to be equal to the standard deviation of the distances computed between every codeword. Given matrix M, we identify the entry $M(x, y)$ in the last row/column which has the minimum value as ϕ. It denotes the misalignment score between the two sequences. The optimum alignment is then found by keeping track of the elements that contribute the minimum distance at each step i.e. backward transversal of a path from $M(x, y)$ to $M(1, 1)$. As illustration, Fig. 10(a) shows an example xSC that is learned for the open-bottle activity, and its alignment to a positive window, where the xSC and window codewords corresponds to different persons. Misalignment score ϕ obtained is given in the same figure. We show the alignment of the same xSC to a random negative window (drinking-water activity) in Fig. 10(b). As observed, a higher misalignment score is obtained when xSC is paired to a negative window.

The formulation of the sequence alignment in the above share similarities to the classical time warping, but has an important difference in that time warping considers the distances of vector pairs taken from a common k-dimensional feature space, whereas sequence alignment considers the distances of codewords taken one each from each sequence [1]. Here, insertion and deletion of gaps in the alignment is similar to lengthening (slowing-down) and shortening (speeding-up) of the activity in the video sequence, and empowers the alignment procedure to be robust to temporal variations of an activity. Additionally, we compute the optimum alignment based on the pose differences between the codewords (rather than demanding the exact matching of codewords). This affords us much flexibility when aligning sequences with slight pose variations. In this aspect, we attempt to strike a winning tradeoff: exploit the appearance and temporal orderings of the codewords encoded in a xSC to empower it with strong discriminative potential to recognize an activity, and leveraging on an alignment framework which affords an xSC with sufficient flexibility to align to test sequences with slight pose and temporal variations.

5.2 Learning discriminative xSC

We learn a boosted ensemble of xSC from positive and negative training sequences for each target activity. These xSC are aligned to test video codeword sequence with a slid-

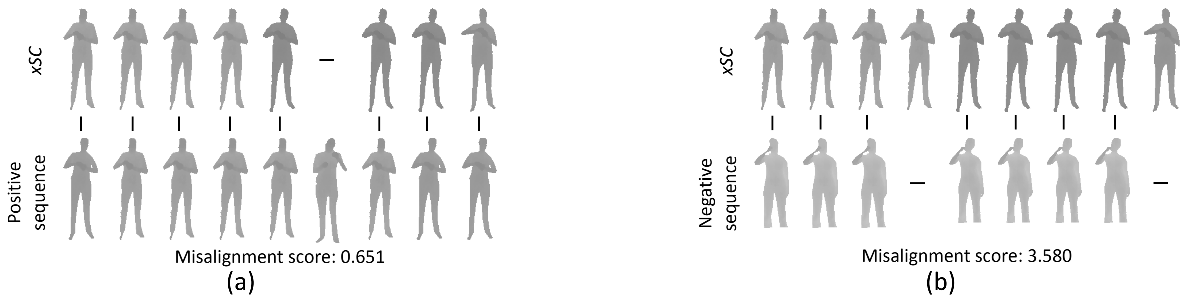


Figure 10: Alignment of example positive and negative window codewords on the same xSC that is learned for the open-container activity. Vertical and horizontal lines between codewords denote the alignment of the codewords and the gaps inserted to obtain the optimum alignment, respectively. Misalignment scores (ϕ) are show for each alignment.

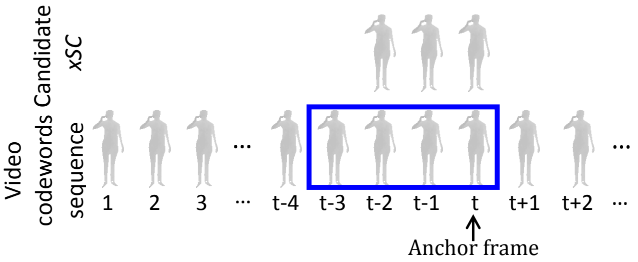


Figure 11: Aligning a candidate xSC to a video codeword sequence. We show the window by the blue rectangle, and also depict the anchoring frame for the window. Candidate xSC assigns a misalignment score to each image frame of the video codeword sequence at the anchoring frame of the window.

ing window approach at inference, where weighted sum of the confidence score at each window is used as the overall confidence measure for the targeted activity at the window. We first present our method to learn a single discriminative xSC, before discussing how a boosted ensemble of discriminative xSC is learned.

We require a discriminative xSC to align well to positive window codewords but not to the negative. In this work, we exploit validation video sequences for discriminative learning of xSC. Specifically, we select a random subsequence of codewords from a positive training codeword sequence as a candidate xSC, and align it to positive and negative validation codeword sequences with a sliding window approach. Rather than assigning a fixed size window for every candidate xSC, we pair each candidate xSC with a window of random size, and anchor each window to a video frame in the validation sequence (see Fig. 11). The anchoring frame provides the reference for the current window. The use of sliding windows enables our method to not only recognize human activities (i.e. classification task), but more importantly empower it with an ability to find a targeted activity within a video sequence (i.e. localization task). We consider the misalignment score (defined as ϕ above) for each window as a feature value of the window, where the class label for the window is +1 if its corresponding anchoring image frame is from a positive validation sequence, and -1 otherwise. The set of feature values and class labels, together with the weights of the windows (initialized according to the number of positive and negative validation video frames) are then used to learn a weak decision stump[13] which maximizes the weighted accuracy on the set of validation windows. The prediction of the decision stump classifier based on the learned xSC at a window w is,

$$xSC\left[w\right] = \begin{cases} +1 & \text{if } \phi < \tau \\ -1 & \text{otherwise} \end{cases}, \tag{11}$$

where τ is a learned parameter. We learn an ensemble of discriminative xSC by AdaBoost [10], where each boosting round outputs an xSC, its weight, window size, and a corresponding decision stump. All xSC learned by AdaBoost are then combined to form a boosted ensemble, where the overall confidence measure of an activity in a window w is

$$H\left[w\right] = \sum_{j=1} \alpha_j \times xSC_j\left[w\right], \tag{12}$$

with α_j as the weight of xSC_j and is learned during boosting. In this paper, we pick candidate xSC by selecting random subsequences from positive training data, and assign its window size a random value between 75% to 125% its length. We sample windows of the validation sequences (both positive and negative) at every frame, and apply 300 boosting rounds to learn the boosted ensemble.

6. HUMAN ACTIVITIES CLASSIFICATION AND LOCALIZATION

In this section, we describe how the learned boosted ensembles are used to recognize human activities (i.e. classification task), and how they are exploited to find human activities within a test video (i.e. localization task). The only image information used by the ensemble is the depth image information, obtained with the Kinect sensor.

To classify the activity depicted in a test video, we learn a boosted ensemble for each activity, and capitalize on these ensembles to recognize the activity-label of the test video. Specifically, we apply the boosted ensemble $H_i\left[\cdot\right]$ which was learned for the i^{th} activity to the test video in a sliding window approach. The sliding step is the same as that used for training. Each window in the test video is assigned a score based on Eq. (12), and we consider the median score of all windows within the test video for the classification confidence of the video as activity i. To localize the i^{th} activity, we apply the boosted ensemble $H_i\left[\cdot\right]$ on all windows of the test video, and employ the powerful mean shift mode estimation technique [7] on the confidence measure of each window, similar to [16]. Mean shift models the non-parametric distribution with the kernel density estimator,

$$P\left(\mathrm{w}\right) \propto \sum_{w_j \in W} H_i\left[w_j\right] G\left(\frac{\mathrm{w} - w_i}{b_w}\right) \qquad (13)$$

where the Gaussian kernel G uses bandwidth b_w. Mean shift efficiently finds modes within the sequence which are then used as the final set of detections. Here, we used the density estimated at each mode as the confidence value of the detection.

7. EXPERIMENTAL RESULTS

7.1 Classification results

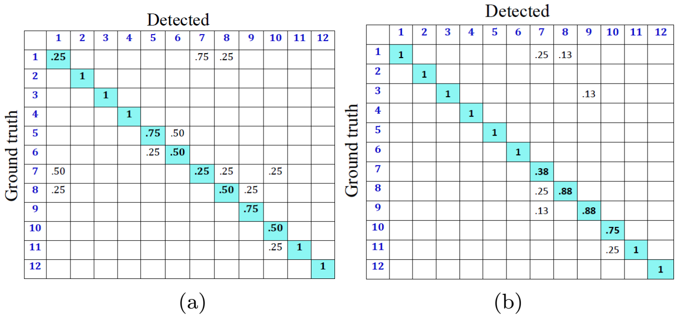
We evaluate our technique on the challenging Human Activity dataset [29] and compare against the best (to our knowledge) classification results obtained so far on this dataset. We used only depth information in our method and adhere to the same evaluation protocols as those used by other methods. This dataset comprises 12 challenging activities (see Table 1) performed by 4 persons. We note that humans in the dataset are located away from the background. Moreover, since the background images are also not available with the dataset, we can not use our proposed method for segmentation. Hence, we apply a simple depth threshold value to segment the human from the background. In our work, we use the Otsu's method [22] to learn an optimum depth threshold value to extract the foreground human. Following [29], we separate the 12 activities into five different environments of office, kitchen, bedroom, bathroom, and living room, and evaluate under two settings: 'new-person' and 'person-seen-before'.

In the 'new-person' setting, we evaluate the boosted ensemble learned for each activity on video sequences in which the ensemble has not previously seen the person carrying out the activity. We used leave-one-out cross validation to test each person data whereby the ensemble was trained on the data of three persons and tested on the fourth person. To learn an ensemble for an activity-class, we pick the positive sequence of one person (randomly chosen) as the training sequence, and use the sequences (both positive and negative) of the other two persons for validation. To evaluate the ensemble, we apply it on data of the person not used for training/validation, in which all activities of the person are used for testing. In this aspect, each activity is searched for in video sequences of *every* activity class. In the 'person-seen-before' setting, we evaluate the boosted ensemble on video sequences for which the ensemble has previously seen the person carrying out the activity. Here, we report two-fold cross validation classification accuracy, in which we split the video sequence of each person into two equal halves, used one half for training/validation, and the other for testing.

We first discuss classification results for the 'new-person' setting. Fig. 12(a) reports the confusion matrix of the activities irrespective to different environments. Turning to incorrect classifications, it can be seen that activities brushing-teeth and talking-phone are often misclassified. This is due to the similar body poses for both activities which is further complicated by little hand motions in both activities. Consequently, our method which exploit the appearance and temporal cues fail to achieve accurate classification on these activities. On the other hand, our method attain high accuracies for activities which have unique poses (e.g. working-on-computer) and unique temporal pose ordering (e.g. rinsing-mouth). We compare results obtained by our method with *Naive Classifier (multi-class support vector machine)*, *One-level MEMM*, *two-level MEMM*[29] in Table 2. Our method is able to detect and classify the activities performed in different environments with an overall average precision/recall measure of 78.1%/75.4% which is better than 67.9%/55.5% obtained by Sung *et al*. [30] under this setting (bottom row of Table 2). It has to be mentioned that [30] exploits the pose information obtained with robust tracking of body joints locations, while our method uses silhouettes that are (crudely) segmented from depth images. This demonstrates the power of our learned ensemble to recognize complex activities.

Table 1: Activities in Human Activity dataset [29].

ID	Activity	ID	Activity
1.	Brushing teeth	7.	Talking on phone
2.	Rinsing mouth	8.	Drinking water
3.	Wearing contact lenses	9.	Opening container
4.	Working on computer	10.	Talking on couch
5.	Cooking - Chopping	11.	Relaxing on couch
6.	Cooking - Stirring	12.	Writing on whiteboard



Detected
Ground truth

	1	2	3	4	5	6	7	8	9	10	11	12
1	.25						.75	.25				
2		1										
3			1									
4				1								
5					.75	.50						
6					.25	.50						
7	.50						.25	.25		.25		
8	.25							.50	.25			
9									.75			
10										.50		
11										.25	1	
12												1

(a)

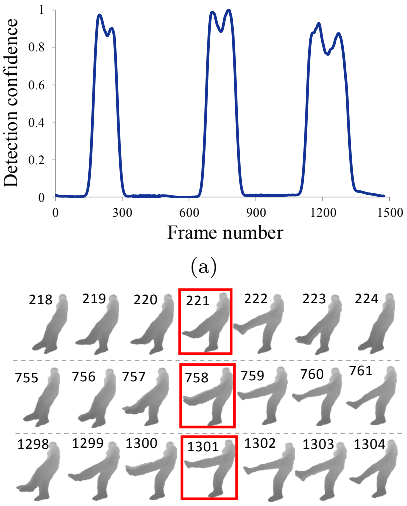
	1	2	3	4	5	6	7	8	9	10	11	12
1	1						.25	.13				
2		1										
3			1						.13			
4				1								
5					1							
6						1						
7							.38					
8							.25	.88				
9							.13		.88			
10										.75		
11										.25	1	
12												1

(b)

Figure 12: Confusion Matrices for each activity are shown in (a) and (b) respectively. Results for "new-person" setting are shown in (a), and "person-seen-before" setting in (b) irrespective to different activity environments. Indices in first row and column of each table corresponds to those in Table 1.

Figure 13: Example sequence of training images used to localize the *left-leg-kicking* activity from a test sequence.

Fig. 12(b) reports the confusion matrix for the 'person-seen-before' setting. It can be seen that the boosted ensemble has sharp improvement in performance. This is not surprising, since this experiment setting allows the boosted ensemble to directly exploit the unique characteristic of the person performing the activity to learn the xSC, but does highlight the capability of our method to hone in on these characteristics. Here, our algorithm is able to detect and classify the activities performed in different environments with an overall average precision/recall measure of 94.6%/ 93.1% which is better than 84.7%/83.2% obtained by Sung *et al*. [30] for the same setting (bottom row of Table 2). To our knowledge, this is the best result published so far for this dataset.



(a)

(b)

Figure 14: Activity localization. (a) Plot of detection confidence against frame number, obtained by applying the boosted ensemble of the test video using the sliding window approach. (b) Detections obtained by our method, where each row correspond to a detection and number indicate frame index. Red bounding boxes within segmented body correspond to detected modes.

7.2 Localization results

In this section, we demonstrate our method to localize a targeted activity from a test sequence. Here, a human is very near to the background (lying on the ground) and hence we use the proposed method to extract the foreground. We define the targeted activity as the kicking motion of the left leg, and show an example training sequence of the activity in Fig. 13. We test our method on a test sequence which shows a person performing the targeted activity at three separate times within the sequence. Specifically, this activity were performed at frames f_{175} to f_{280}, f_{676} to f_{847} and f_{1120} to f_{1325}. Frames f_1 to f_{174}, f_{281} to f_{675}, f_{848} to f_{1119} and f_{1326} to f_{1478} comprise random activities perform by the person. We show example frames from each interval in Fig. 15. We learn a boosted ensemble of xSC for the target activity, and plot the detection confidence for each sliding window in Fig. 14(a). Detections obtained by mean shift mode estimation corresponds to the three peaks in the graph. We show these detections in Fig. 14(b), where the fourth frame in each detection corresponds to the detected modes. These detections accurately localize the left-leg-kicking activity, and demonstrate the activity localization capability of our method.

8. CONCLUSION

We presented a method which uses only depth images for activity classification and localization. Towards this end, we proposed a descriptor to represent a human pose, and exploited a spatial-based clustering method to find unique human poses (i.e. codewords). Given a training video, we represent the video as a sequence of codewords and learn subsequences of codewords (termed xSC) that model both the discriminative poses of a human when performing an activity, as well as the distinctive temporal ordering of these poses. We identify such a discriminative xSC as one which aligns well only in positive test videos. Here, we formulate the problem of alignment of xSC with a test window codewords as the DNA alignment problem, and exploit the semi-global alignment method to find the optimum alignment. The insertion/deletion of gaps in the alignment represents the slowing-down/speeding-up of the targeted activity in the videos, and affords the alignment to be robust towards temporal variations in the activity. At the same time, robustness to pose variations is achieved since the optimum alignment is computed from the pose differences of the codewords (rather than demanding the exact matching of codewords). Activity classification and localization results on test videos demonstrate the effectiveness of our approach, in which for the Human Activity dataset, we achieved improvement over the best results published so far for the dataset.

Acknowledgements: We thank the anonymous reviewers for their valuable input. This work is supported by Institute for Infocomm Research and by Microsoft Research Asia.

9. REFERENCES

[1] J. Aach and G. M. Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508, 2001.

[2] J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43, 2011.

[3] T. Attwood and D. Parry-Smith. Introduction to bioinformatics, 1999. Addison Wesley Longman.

[4] P. J. Besl. *Surface in Range Image Understanding.* Springer-Verlag New York, Inc. New York, 1989.

[5] S. T. Birchfield and S. Rangarajan. Spatiograms versus histograms for region-based tracking. *CVPR*, 2005.

[6] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *Trans. on Pattern Analysis and Machine Intel.*, 23:257–267, 2001.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Trans. on Pattern Analysis and Machine Intel.*, 24(5), 2002.

[8] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. *In: ICCCN, IEEE*, pages 65–72, 2005.

[9] A. A. Efros, A. C. Berg, E. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. *ICCV*, 2003.

[10] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Comp. and System Sc.*, 55:119–139, 1997.

[11] S. Holzer, J. Shotton, and P. Kohli. Learning to efficiently detect repeatable interest points in depth data. *Eurpoean Conf. on Comp. Vision*, 2012.

[12] M.-K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. on Info. Theory*, 8, 1962.

[13] W. Iba and P. Langley. Induction of one-level decision trees. *Int. Workshop on Machine Learning*, 1992.

[14] S. Kumar and A. Filipski. Multiple sequence alignment: In pursuit of homologous dna positions. *Genome Research*, pages 127–135, 2007.

[15] I. Laptev. On space-time interest points. *Int. Journal on Computer Vision*, 64:107–123, 2005.

[16] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. *Eurpoean Conf. on Comp. Vision Workshop*, pages 17–32, 2004.

[17] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. *Comp. Vision and Pattern Recognition workshops*, 2010.

[18] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. *ICCV*, 2003.

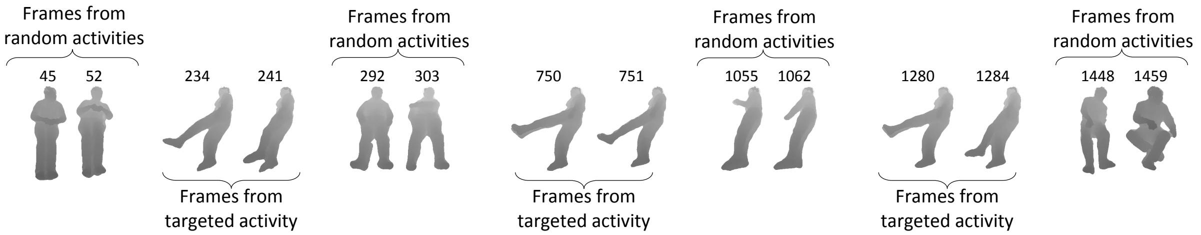

Frames from random activities
45
52
234
241
Frames from targeted activity
Frames from random activities
292
303
750
751
Frames from targeted activity
Frames from random activities
1055
1062
1280
1284
Frames from targeted activity
Frames from random activities
1448
1459

Figure 15: Example frames from different intervals of the test video sequence. Intervals shown belong to both targeted and random activities.

Table 2: ***Precision*** **and** ***Recall*** **scores of** ***Naive Classifier (multi-class support vector machine)*****,** ***One-layer MEMM*** **model,** ***Two-layer MEMM*** **model, and** ***Our method*** **in each environment.**

Location	**Activity**	**new-person**								**person-seen-before**							
		Naive Cl-assifier		One-layer MEMM		Two-layer MEMM		Our Method		Naive Cl-assifier		One-layer MEMM		Two-layer MEMM		Our Method	
		Prec	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*	*Prec*	*Rec*
bathroom	rinsing mouth	77.7	49.3	71.8	63.2	51.1	51.4	**100**	**100**	73.3	49.7	70.7	53.1	61.4	70.9	**100**	**100**
	brushing teeth	64.5	20.5	83.3	57.7	88.5	55.3	**100**	**75.0**	81.5	65.1	81.5	75.6	96.7	77.1	**100**	**100**
	wearing contact lens	82.0	89.7	81.5	89.7	78.6	88.3	**80.0**	**100**	87.8	71.9	87.8	71.9	79.2	94.7	**100**	**100**
	Average	**74.7**	**53.1**	**78.9**	**70.2**	**72.7**	**65.0**	**93.3**	**91.6**	**80.9**	**62.2**	**80.0**	**66.9**	**79.1**	**90.9**	**100**	**100**
bedroom	talking on phone	82.2	32.6	82.0	32.6	63.2	48.3	**60.0**	**75.0**	70.2	67.2	70.2	69.0	88.7	90.8	**100**	**62.5**
	drinking water	19.2	12.1	19.1	12.1	70.0	71.7	**50.0**	**50.0**	64.1	31.6	64.1	39.6	83.3	81.7	**80.0**	**100**
	opening container	95.6	65.9	95.6	65.9	95.0	57.4	**100**	**75.0**	48.7	52.3	48.7	54.8	93.3	77.4	**88.8**	**100**
	Average	**65.6**	**36.9**	**65.6**	**36.9**	**76.1**	**59.2**	**70.0**	**66.7**	**61.0**	**50.4**	**61.0**	**54.5**	**88.4**	**83.3**	**89.6**	**87.5**
Kitchen	cooking (chopping)	33.3	56.9	33.2	57.4	45.6	43.3	**60.0**	**75.0**	78.9	28.9	78.9	29.0	70.3	85.7	**100**	**100**
	cooking (stirring)	44.2	29.3	45.6	31.4	24.8	17.7	**50.0**	**50.0**	44.6	45.8	44.6	45.8	74.3	47.3	**100**	**100**
	drinking water	72.5	21.3	71.6	23.9	95.4	75.3	**75.0**	**75.0**	52.2	51.5	52.2	52.4	88.8	86.8	**100**	**100**
	opening container	76.9	6.20	75.8	6.20	91.9	55.2	**100**	**75.0**	17.9	62.4	17.9	62.4	91.0	77.4	**100**	**100**
	Average	**56.8**	**28.4**	**56.6**	**29.7**	**64.4**	**47.9**	**71.2**	**68.7**	**48.4**	**47.2**	**48.4**	**47.4**	**81.1**	**74.3**	**100**	**100**
living room	talking on phone	69.7	0.90	83.3	25.0	51.5	48.5	**50.0**	**75.0**	34.1	67.7	34.1	67.7	88.8	90.6	**100**	**75.0**
	drinking water	57.1	53.1	52.8	55.8	54.3	69.3	**66.7**	**50.0**	80.2	48.7	71.0	53.8	80.2	82.6	**80.0**	**100**
	talking on couch	71.5	35.4	57.4	91.3	73.2	43.7	**100**	**50.0**	91.4	50.7	91.4	50.7	98.8	94.7	**100**	**62.5**
	relaxing on couch	97.2	76.4	95.8	78.6	31.3	21.1	**80.0**	**100**	95.7	96.5	95.7	96.5	86.8	82.7	**72.7**	**100**
	Average	**73.9**	**41.5**	**72.3**	**62.7**	**52.6**	**45.7**	**74.2**	**68.7**	**75.4**	**65.9**	**73.1**	**67.2**	**88.7**	**87.7**	**88.2**	**84.4**
office	talking on phone	60.5	31.0	60.6	31.5	69.4	48.2	**60.0**	**75.0**	80.4	52.2	80.4	52.2	87.6	92.0	**100**	**75.0**
	writing on whiteboard	47.1	73.3	45.2	74.1	75.5	81.3	**100**	**100**	42.5	59.3	42.5	59.3	85.5	91.9	**100**	**100**
	drinking water	41.1	12.4	51.2	23.2	67.1	68.8	**66.7**	**50.0**	53.4	36.7	53.4	36.7	82.3	81.5	**80.0**	**100**
	working on computer	93.5	76.8	93.5	76.8	83.4	40.7	**100**	**100**	89.2	69.3	89.2	69.3	89.5	93.8	**100**	**100**
	Average	60.5	48.4	62.6	51.4	73.8	59.8	**81.7**	**81.2**	66.4	54.4	66.4	54.4	86.2	89.8	**95.0**	**93.7**
Overall Average		**66.3**	**41.7**	**67.2**	**50.2**	**67.9**	**55.5**	**78.1**	**75.4**	**66.4**	**56.0**	**65.8**	**58.1**	**84.7**	**83.2**	**94.6**	**93.1**

[19] B. Ni, G. Wang, and P. Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. *Int. Conf. on Comp. Vision Workshops*, 2011.

[20] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. *ICCV*, pages 1–8, 2007.

[21] C. O'Conaire, N. E. O'Connor, and A. F. Smeaton. An improved spatiogram similarity measure for robust object localisation. *ICASSP*, pages 15–20, 2007.

[22] N. Otsu. A threshold selection method from gray-level histograms. *Trans. on Sys., Man and Cyber.*, 9(1), 1975.

[23] V. Parameswaran and R. Chellappa. View invariance for human action recognition. *Int. J. on Comp. Vision*, 66:83–101, 2006.

[24] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. *Intelligent Robots and Systems*, 2008.

[25] B. Sabata, F. Arman, and J. K. Aggarwal. Segmentation of 3d range images using pyramidal data structures. *Int. Conf. on Computer Vision*, 1990.

[26] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. *ICPR*, pages 32–36, 2004.

[27] J. Shotton, A. FItzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. *CVPR*, 2011.

[28] P. Spagnolo, T. Orazio, M. Leo, and A. Distante. Moving object segmentation by background subtraction and temporal analysis. *Image and Vision Computing*, 24(5):411–423, 2006.

[29] J. Sung, C. Ponce, B. Selman, and A. Saxena. Human activity detection from rgbd images. *AAAI workshop on Pattern, Activity and Intent Recognition*, 2011.

[30] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from rgbd images. *Int. Conf. on Robotics and Automation*, 2012.

[31] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. *CVPR*, 2012.

[32] A. Yilmaz and M. Shah. Actions sketch: A novel action representation. *CVPR*, 1:984–989, 2005.

[33] H. Zhang and L. E. Parker. 4-dimensional local spatio-temporal features for human activity recognition. *Int. Conf. on Intelligent Robots and Systems*, 2011.

[34] Y. Zhao, Z. Liu, L. Yang, and H. Cheng. Combining rgb and depth map features for human activity recognition. *APSIPA ASC*, 2012.

[35] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. *ICPR*, pages 28–31, 2004.