# Integrating local Action Elements using Implicit Shape Model for Action Matching

Tuan Hue Thi[12] Jian Zhang[12]

[1] University of New South Wales, Australia
[2] National ICT of Australia
{TuanHue.Thi,Jian.Zhang}@nicta.com.au

**Technical Report**
**UNSW-CSE-TR-1012**
**March 2010**


School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia

**Abstract**

In this paper we propose an approach for analyzing human action in video, and demonstrate its application to several related tasks: action retrieval, action classification and action localization. In our work, actions are represented as a set of local space time features, or Action Elements, then an Implicit Shape Model of the action is built to integrate spatial and temporal correlations of these local Action Elements. In particular, we propose *two* different approaches to extracting Action Elements: a *discriminative* and a *generative* approach, respectively. Action Elements detected from either of the approaches are then used to build an Implicit Shape Model for current action of interest. We apply our action matching algorithm to the application of action recognition and carry out thorough experiments on the two well-known action datasets, KTH and Weizmann. Results obtained from those experiments demonstrate the highly competitive performance of our action matching algorithm compared to state-of-the-arts, and also give evaluative insights on the two proposed feature extraction approaches.

# 1 Introduction

Action analysis earns its great interest from many potential applications. In video mining, there have been a number of practical systems developed to extract the meaningful semantics of image and video content. Those interesting applications can be seen in the Video Google object matching demonstration [43], the landmark structure localization in [7], and landmark three-dimension reconstruction from [44]. Human action analysis and action retrieval in particular, on the other hand, has yet to see successful systems developed for practical usage. In this paper, we present a generic framework for matching actions in video, which can be used as a core module for development of a complete visual-based action search engine.

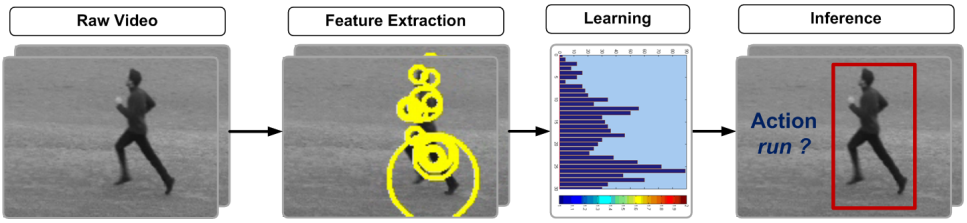



Figure 1.1: Common Framework of a Human Action Recognition System

A common theme for human action retrieval can be decomposed into two main stages, the first concerns with feature extraction and the second deals with learning and inference, as illustrated in Figure 1.1. Existing feature extraction methods can be roughly categorized into two main regimes, one deals with the global feature of the content of interest, while the other uses a set of local features. In our work, we approach video action matching using salient local features, which we denote as Action Elements, and aim to learn not only their local statistics but the implicit global shape. In Section 3, we describe two independent approaches towards extracting those Action Elements, one discriminatively learn those regions that explicitly distinct one action class from others, while the second approach apply a cascaded generative filters to depict the most compact elements of an action instance.

By representing video as different sets of sparse features, and embed standard local attributes in their transparent representation, we can apply different model learning techniques to recognize and match similar action patterns. In Section 2, we give a more detailed revision on works related to our approach. Section 3 first denote our representation of video action, defining sparse sets of local features as Action Elements and describing them using rich and distinctive local information over space and time. Using this representation, we then present two effective approaches for extracting Action Elements from raw video. The first approach uses a discriminative Bayesisan classifier to determine if a detected Space Time Interest Point should be included into Action Element set. The second technique combines the advantage of holistic features via MHI value and affine invariant local regions using Hessian-Laplace detector. The detected regions are then passed through a generative kernel that use local motion and shape information, as well as self-matching of these points to pick only the most

compact local features. At the end of Section 3, we have two distinctive way to transform raw video into meaningful Action Elements with their embedded rich local information.

Section 4 will integrate those Action Elements into a dynamic structure, described by the mutual relationship among them. Specifically, we decompose action matching task of two video actions into two stages. First, we use build a Hough space model from an action, then we project the other action onto this space. The matching problem now becomes evaluating the fitting of projected space. In Section 4.3, we give a detailed walkthough of constructing a probabilistic matching function for two video actions. We then describe our way of utilizing the geometric characteristic to solve the multi-dimensional density estimation. At the end of this section, we also introduce another function for finding mutual distance of any two video actions. The two functions together help us to solve practical applications of action retrieval and action classification. Section 5 gives a comprehensive evaluation of our matching framework on the two datasets KTH and Weizmann. Using different experimental results on different types of actions, we then draw some useful conclusion about the distinctive difference between the two local feature extraction and their potential fields of application. We also include in the Appendix a simple demonstration of the self-fitting case in voting approach and some qualification evaluation of action element extraction.

## 2 Related Works

This work is related to several domains of feature extractions and learning techniques. We first explore two approaches to detect and select local interest features, specifically developed to represent the distinctiveness of each action class, in our design, we call those salient local patches as Action Elements. The first approach starts with the local spatial temporal interest points detected directly on the raw video shots using Space-Time Interest Points from [24]. These interest points will be passed through a Sparse Bayesian 3D Feature Classifier that can filter those representative features of the action, this technique is an extension from 2D object feature categorization of [5]. The second approach starts with the holistic feature of Motion History Image developed by [4], we then apply affine-invariant feature transform using Hessian-Laplace region detector and SIFT [31] descriptor from [32], to select the interesting local regions. Those feature candidates are then passed through our generative feature filter to select only the most characteristic features based on their shape and motion behavior. Our first feature extraction approach can be understood as a discriminative way to build an action feature filter that picks only those features that directly contribute to the actions. Meanwhile, the second approach generatively learns the most distinctive local features detected on the motion field. From our empirical experiments, the first approach is more suitable for the task of action classification where more training features are available, while the latter suits best to our interest in the task of action search where normally only one query action is given, formal evaluation of these two approaches will be shown in Section 5.

Vision model learning techniques can be roughly categorized into generative, discriminative and voting-hashing based. Generative models are popular in object recognition, especially for matching similar object categories

[15][42][13][45][34], and recently similar techniques have been adapted to action recognition [36][35]. The advantage of generative approach is with its ability to learn the representative characteristics of action and can be used to infer on unknown actions. The discriminative approach, on the other hand, explicitly explore the distinction of a action from the others, and use that knowledge to learn the model, works in object recognition can be seen in [8][16][56][14], where extension in action recognition is shown in [50] or our previous work on integrating human body parts into a dynamic graphical model [47]. Although these works are reported with good results in action recognition, it is still a limitation that the structure of each action is not well depicted. That motivates the recent researches in voting and hashing techniques for object recognition [29][17][37][41][20][48], the core idea is to project sparse local features onto a common space that roughly represents the shape of the global configuration of these features as well. These works have been proven successful in many object recognition tasks, from categorization to matching, and visual searching.

In our work, we develop an Implicit Shape Model (ISM), inspired by Object Categorization approach in [29], to combine extracted Action Elements with their global configuration . We build a generalized Hough Space [2] using one video action as the model. The task of action matching now becomes projecting a new video action using this model space, and the matching distance is estimated from the fitting performance. Figure 2.1 summarizes the overview of our action matching framework, specific part will be explained in details throughout the rest of this paper.

In the discriminative feature extraction approach, we build the Action Elements from the Space-Time Interest Point proposed in [24], which basically extend the Harris corner detector to three dimensions and use a concatenation on Histogram of Oriented Gradients (HoG) and Histogram of Oriented Flow (HoF) [9] to describe each interest cube. Common approaches put these detected points directly into either a discriminative Support Vector Machine [40][27][25] or use Bag-of-Word approach to build a feature codebook entry for behavior analysis [10][36]. Although these approaches are reported to achieve high performance in the action recognition task, they typically neglect the global configuration of the action itself, solely relying on separate sparse features distribution to infer on the action, hence, they are highly context-sensitive and less generative for learning other instances of the same action class. Observing direct use of STIP cannot give us the most representative yet generic elements for action class, we adopt an additional discriminative filter that helps to weigh interest points according to its contribution to the action learning. This Bayesian feature filter is extended from the work of [5] and [6] in Object Recognition task. The core idea behind this feature selector is with the formulation of a conditional distribution that combines sparsity learning with feature selection regression of each feature type, which can be used to retrieve important features from detected candidates. Extending this notion into the selection task of video features, we can then be able to discriminatively build an action feature labeler that probabilistically judges if a local feature is contributing to the action behavior. Those positively classified features are the Action Elements of our interest.

Apart from the discriminative feature extractor, we also propose a generative approach to retrieve compact action elements using information fusion of holistic feature and local descriptor. This approach is somewhat similar to the work of [30] which detects SIFT [31] local regions from global energy image [4].

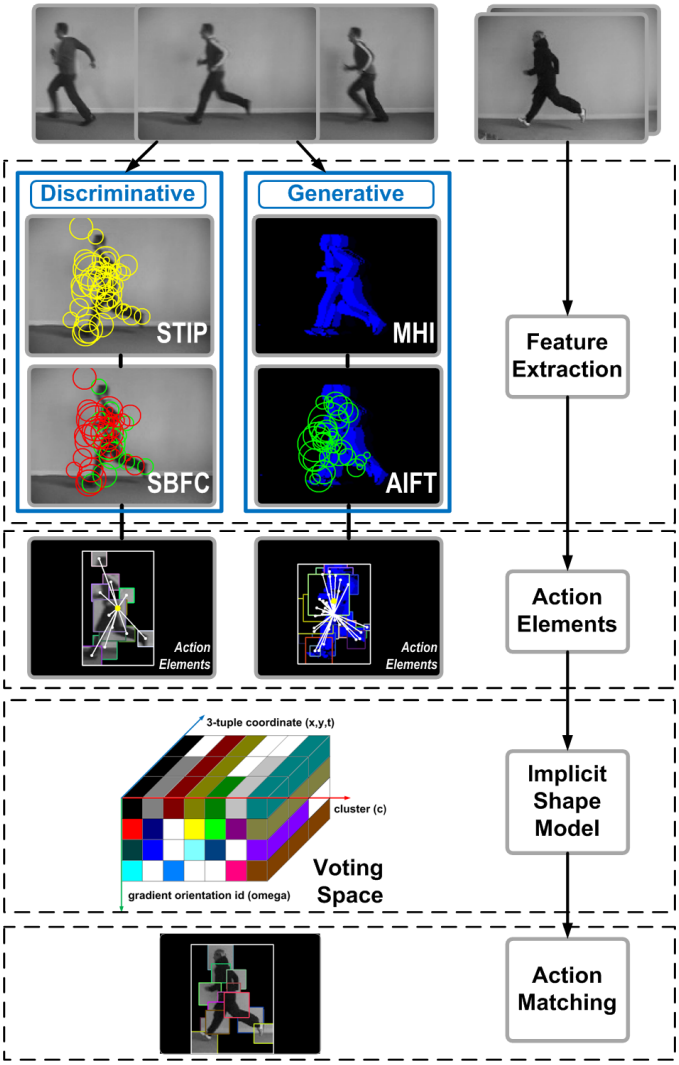



Figure 2.1: Our Action Matching Framework, starting with query action on the left, there are two ways to extract Action Elements, either with Discriminative approach using Space Time Interest Point (STIP) and Hierarchical Bayesian Kernel Machine (HBKM) or with Generative approach using Motion History Image (MHI) and Affine Invariant Feature Transform (AIFT). Action Elements are used accordingly to fill the Voting Space of the Implicit Shape Model. Unknown video shot on the right is passed through the same process to find the Action Element candidates, which are then fitted into the filled Voting Space to derive the matching score as well as segmented Action Elements

We employ the interesting notion of their work in combining holistic motion field with scale invariant local features, and introduce an additional generative kernel filter to select features based on their shape,motion, and self-matching characteristics, more details about this filter will be given in Section 3.3. The

initial motivation for this approach is to have a generative way that only uses a query video action of interest as the training source to detect the most salient local motion regions of the raw video as use them as the Action Elements for the model construction. This feature extraction begins with the Motion History Image (MHI) constructed using the same method described in [4]. MHI is basically an image built from the pixel difference of consecutive frames, these images are representative for the motion field and their chronological order detected from the raw video. Common action recognition applications of this holistic-feature MHI are normally developed to take the whole MHI silhouette and build up the 3D action global shape, which then convert action recognition problem into a three-dimension object categorization task [28][11][49][3][54]. The common drawback of these global feature based approaches is caused by their high dependency on good foreground extraction. This problem makes these approaches less practical in the case of video action recognition where usually many noisy factors are also included along with the action regions of interest. In our approach, we are interested in utilizing the motion field to build up the action element codewords, and also try to avoid the inconvenient drawback of global features, we articulate the MHIs into small local regions using affine-invariant Hessian-Laplace interest region detector from [32] and describe them using Scale Invariant Feature Transform descriptor from [31]. Those features are then further characterized using an additional generative kernel to select only those that compactly represent the motion in the video shot. Those local features that can successfully pass this step are then fed into our final Action Element set, similar to the previous discriminative approach.

Having obtained the compact set of Action Elements generated either one of the two feature extraction approaches, we then integrate their local characteristics with global configuration constraints into an Implicit Action Shape Model (ISM). This voting-space model is motivated from the work of [29] in 2D Object Categorization. Specifically developed for 3D action learning, our ISM characterize local identities with probabilistic awareness of the integral global structure. We implement this ISM using a Generalized Hough Transform based on the work of [2] to construct a Hough Space that conveys the geometric notion of the action based on each individual contribution of the Action Elements. These applications of Hough Transform was initially described in [28][31][29] that built different Hough Space models for the purpose of recognizing arbitrary shapes with [2], gait poses recognition in [28], and object recognition applications in [31] and [29]. To our observation, there has been no reported work conducting using Hough Transform on local features for action recognition, and one of the possible reason might be that Hough Transform has large error bound for estimation, which makes it highly sensitive to noisy video data. In our approach, we effectively cope with this challenge using a sophisticated feature preprocessing to retrieve only the most concise Action Elements of the action model, as described in Section 3.2 and 3.3. Using the Hough Space generated from one query action, we can then apply the same preprocessing procedure to detect possible Action Elements from unknown action, and project them onto the model Hough Space. Action matching score is then conceptually estimated as how well the projected parameters fit into the model.

We have three main contribution in this work. Firstly, we propose and evaluate two new techniques to extract salient local features. Secondly, we develop an Implicit Action Shape Model to combine their global configuration with local

information, together with two probabilistic functions for action matching and action distance. Thirdly, we carry out comprehensive evaluation of the proposed matching algorithm on the particular task of action retrieval and action classification.

# 3 Action Elements for Local Representation

We tackle the problem of Action Recognition in video using local feature-based approach. That is, we aim to extract the most representative local features from a raw video, and use knowledge obtained from them to infer on the whole action behavior. In this section, we describe our way of representing video action, and develop two approaches for local feature extraction from raw videos. The first approach builds upon the Space-Time Interest Point (STIP) [24] to extract the spatial-temporal local features and uses Sparse Bayesian Feature Classifier (SBFC) [5] to select only those that help to distinct the action from other classes. The Feature Classifier is discriminative, so we roughly call this technique Discriminative Feature Extraction, as opposed to the second approach, Generative Feature Extraction. The second technique aims to find the most representative local features of a video source using the combination of holistic features Motion History Image (MHI) [4] and Affine Invariant Feature Transform (AIFT) for local feature extraction. The detected features are then passed through a generative model of different filters to select those that are most representative for the action.

## 3.1 Video Action Representation

Normally, visual information of a video $\mathcal{V}$ is defined by a collection of its pixels $I$, that is $\mathcal{V} \supset I(x, y, t, i)$ with coordinates $(x, y, t)$ and intensity $i$. We approach video action in an analogous way, decomposing an action $\mathcal{A}$ into local salient patches, which we roughly name as Action Elements $\mathcal{AE}$. Within each $\mathcal{AE}$ there can be a range of information that can be embedded from the video domain, and together, all these elements will depict the distinctive global shape of the underlying action. Based on this observation, the action recognition task now becomes the problem of extracting salient local $\mathcal{AE}$ and learning their implicit global shape.

An $\mathcal{AE}$ in our design is defined by a feature vector $(x, y, t, s, d, c, \omega)$, where $x$, $y$, and $t$ indicate the geometric position of $\mathcal{AE}$ patch center, $s$ specifies its scale in region radius, $d$ is the feature description, $c \in \mathcal{C}$ defines the feature description cluster, and $\omega \in \Omega$ represents the dominant gradient orientation. In this notation, $\mathcal{C} = \{< c, d >\}$ is the set of all feature description clusters, defined as a pair set of cluster identification number $c$ and average description vector value $d$, more details about the feature descriptor and cluster techniques will be described in 4.1. Meanwhile, $\Omega$ is the set of gradient orientations, obtained from the averaging of all feature description $d$ gradients in each $\mathcal{AE}$. By including the two values $c$ and $\omega$, we have embedded local shape and motion behavior from the raw video into our $\mathcal{AE}$. Our feature extraction steps in Section 3.2 and 3.3 will select and explain the type and meaning of $c$ and $\omega$.

For the rest of the paper, we will use this notation $\mathcal{A}_u \supset \mathcal{AE}^u(x, y, t, s, d, c, \omega)$ to represent a video action $\mathcal{A}_u$ obtained from video shot $\mathcal{V}_u$. Figure 3.1 illustrates

our representation of action *boxing* as a set of $\mathcal{AE}$.

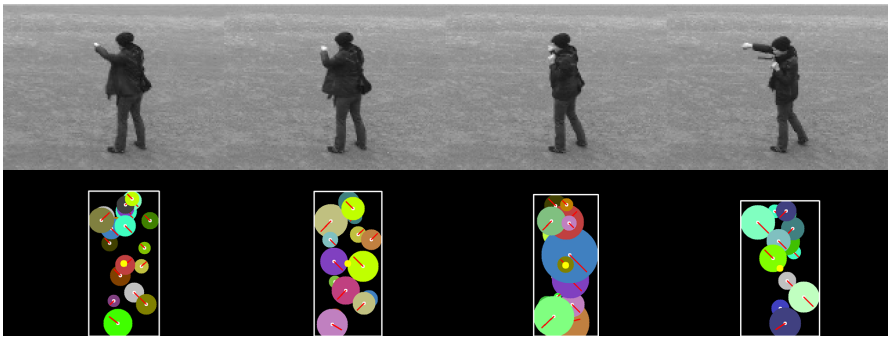


Figure 3.1: Action Elements $\mathcal{AE}$ for KTH [40] action *boxing*: each $\mathcal{AE}$ is represented as a filled circle with its relative location $(x, y, t)$, their colors indicate different cluster identification $c$, and the red line inside each $\mathcal{AE}$ shows the dominant gradient orientation

## 3.2 Discriminative Action Elements Extraction

In action recognition, the common problem with local feature detectors is the inclusion of many noisy and irrelevant features in the outputs, which increases complexity of the learning model and reduces recognition performance. This feature extraction and selection technique is motivated from the work of local feature labeler for object categorization by [6], which uses a Bayesian kernel machine to determine if the a feature actually belongs to the object of interest. We extend their discriminative feature classification model to our case of action recognition using three dimensional features and incorporate motion constraints to extract only those that are distinctive for the action.

### Space-Time Interest Points Detection

For this particular approach, we use Space-Time Interest Points (STIP) from [24] to extract local spatial and temporal feature cubes from video. This interest point detector technique looks for highly variant local regions in a video where the change in both space and time is significant. Empirically, we observe that STIP only works effectively when the appropriate parameters are chosen, depending on the context of the video source. In order to ensure STIP stability, we include a preprocessing step to estimate the relative motion measure of each actor in the video context and pick the appropriate parameters for STIP. Figure 3.2 shows the example STIP detection of Weizmann action *pjump*. Note that, at this initial detection stage, we aim to include all possible salient regions of the action, at the expense of noisy feature inclusion, which will be filtered later using Bayesian classifier.

### Sparse Bayesian Feature Classifier

The advantage of choosing Sparse Bayesian Feature Classifier (SBFC) model [5] for this task if three-fold, firstly it can represent the sparsity of detected STIPs,

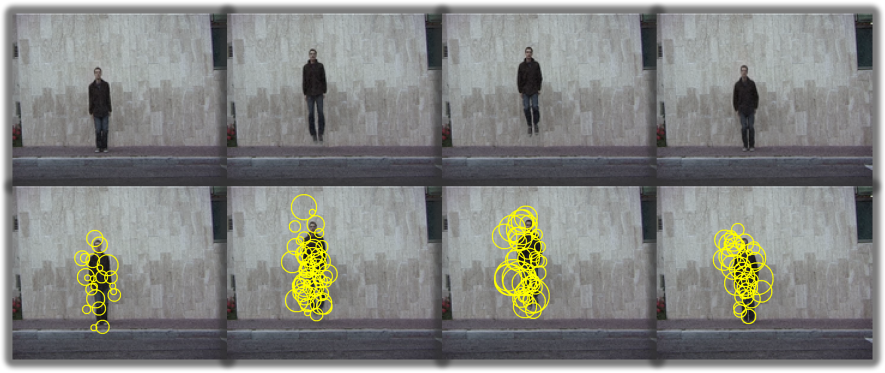


Figure 3.2: Our adaptive STIP detection of Weizmann [18] action *pjump*: Yellow circles with different size are the detected interest regions, at this point few noisy features are also included, but our purpose is to sufficiently cover the action region

secondly is its ability to associate feature context statistics, in our case we feed in the proportion of true features and color variance, and thirdly, it can operate from a small number of training data, which is of our particular interest. The discriminative model of two dimensional feature classifier in [5] can be extended to our spatial-temporal features as

$$p(Y|X,\beta,\gamma) = \Phi(f(X,\beta,\gamma)) \tag{3.1}$$

with X is the observed local features, and Y is feature labels

$$Y = \begin{cases} 1 & \text{if } f(X,\beta,\gamma) > 0,\ X \text{ is selected as true Action Element} \\ -1 & \text{otherwise} \end{cases} \tag{3.2}$$

and $\Phi$ as the model probit link [5], $f(X,\beta,\gamma)$ is the regression function with coefficients $\beta$ and sparsity $\gamma$ of the detected STIP. We apply the same learning procedure as they describe with Bayesian rule to classify new features $(Y', X')$ using trained data

$$p(Y'|X', X, Y) = \iint p(Y'|X', \beta, \gamma) p(\beta, \gamma|X, Y) d\beta d\gamma \tag{3.3}$$

In addition, we also include motion variance of categorized STIP feature groups as additional data association into this discriminative model. For the case of action matching, we use background statistics as training data; with action classification task, as described later in Section 5.4, feature instances from the same action class is used. Our adopted model for action feature classification works particular well for two datasets KTH [40] and Weizmann [18], as shown in Figure 3.3

Those positively labeled features, the green circles in Figure 3.3, are chosen as the final Action Elements $\mathcal{AE}$ and used for model learning and matching in Section 4. As suggested by [26] for best performance with extracted STIP

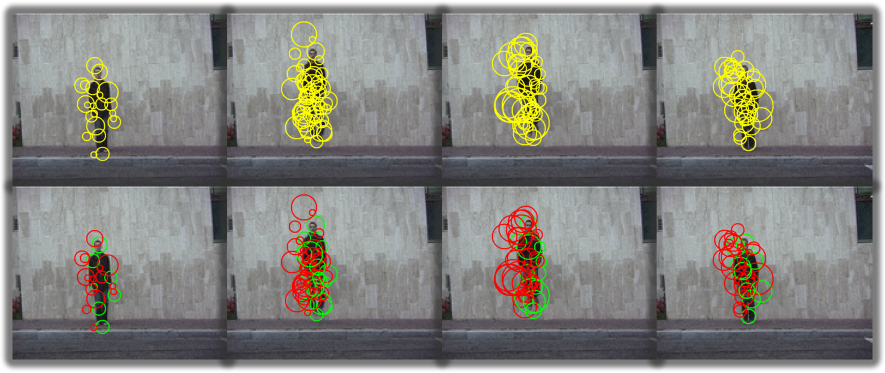


Figure 3.3: Sparse Bayesian Feature Classification result for STIPs detected in Figure 3.2. Red circles are noisy features from background, Green circles are the actual action features

features, we use a concatenation of rectangular Histogram of oriented Gradient (HoG) and Histogram of oriented (HoF) [9] to describe our feature vector detected at each patch, in other words, $d$ is specified by a vector of $HoG - HoF$ values, $\omega$ is directly obtained from HoG in $d$.

Since these $\mathcal{AE}$ are extracted using this discriminative-based approach, they are named Discriminative Action Elements $\mathcal{DAE}$ as opposed to Generative Action Elements $\mathcal{GAE}$, which will be described in the next section. The effectiveness of this Discriminative Action Element Extraction technique will also be quantitatively evaluated in Section 5.

## 3.3 Generative Approach

In this section, we will introduce another approach for extracting action elements from source video. This approach is inspired by the application potentials of visual-based action searching, where there is normally only one query action is provided for learning. We develop a framework to combine global holistic with local features, then use a generative filter function to robustly select those local features that are salient for each kind of action.

We are particularly interested in motion and shape distribution at each local region, also the ability to detect the motion region regardless of the clutter background. To these ends, we use Motion History Image (MHI) [4] technique to extract motion change from source video and concatenate them using their timestamps. The main distinction of our approach compared to traditional holistic-based approaches is that in our design, we do not use one whole MHI to recognize action, but instead we collect motion changes over short time intervals and integrating detected local descriptors for action recognition. Similar to the adaptive step with STIP detection, we also use preprocessing motion estimation to select appropriate parameters for MHI calculation. The second row of Figure 3.4 illustrates those MHI snapshots we calculate for KTH [40] action *walking*.

The last row of Figure 3.4 shows detected local regions on MHI using Affine Invariant Feature Transform Hessian-Laplace interest point detector. Among those detected points there are usually included noisy features from the moving

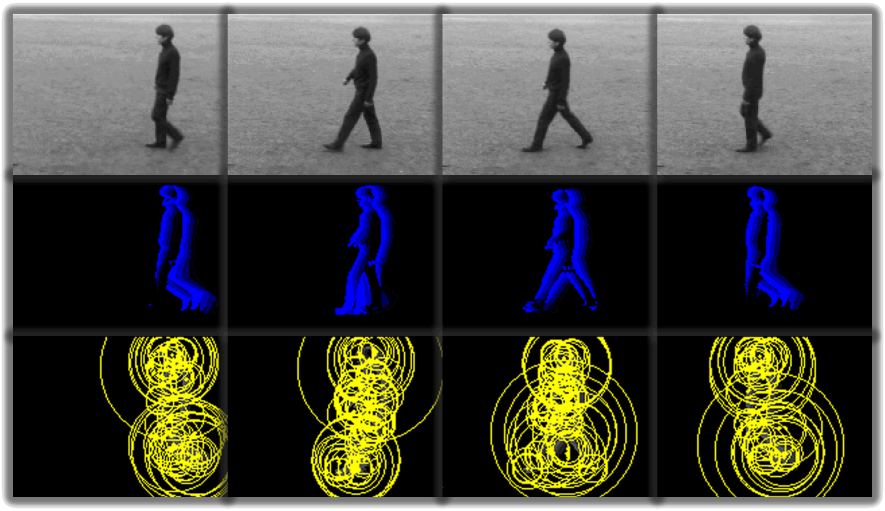


Figure 3.4: Fusion of holistic MHI [4] features with local Hessian-Laplace [32] descriptor on KTH [40] action *walking*. The second row shows the detected results for adaptive MHI, their intensities reflect motion change and timestamp history. The third row illustrates the local feature detection on these MHI using Hessian-Laplace interest point

background, those are the points that have large variation in scale and appearance. We apply a generative kernel $p(X, Y)$ to filter out those noisy features based on scale factor $X_S$ and feature-matching function $m(X)$, normalized by a standard Gaussian $G(\sigma^2)$

$$p(X, Y) = f(X_S, m(X)) * G(\sigma^2) \tag{3.4}$$

with $f$ as weighting function between feature scale and its feature description matching behavior. In this notation, $Y$ is still the decision variable for whether feature $X$ should be included in the final action feature set. The feature-matching function $m(X)$ is developed similar to the idea of match filter for object recognition, described in [31], to evaluate on the ratio of best-match and second-best-match. In addition, we also carry out time matching to find the self-matching distribution over time, Figure 3.5 illustrates feature-matching function $m(X)$

Those features that pass all the filters are the final Generative Action Elements $\mathcal{GAE}$. In this approach, we use Scale Invariant Feature Transform (SIFT) as local descriptor for $d$ which also makes it easy to find the most dominant gradient orientation $\omega$ in each $\mathcal{GAE}$.

# 4 Implicit Global Shape Model of Action Elements

At this point, we have two approaches for extracting $\mathcal{AE}$ from raw video. Traditional local feature-based approaches will put these $\mathcal{AE}$ directly into a discrimi-

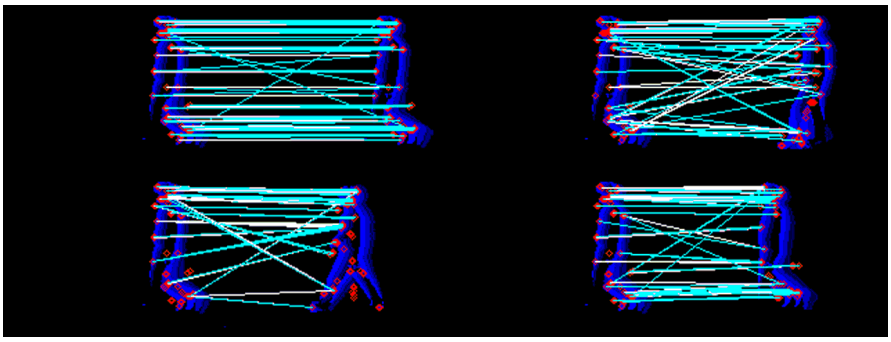


Figure 3.5: Feature-matching procedure for filtering detected feature in generative approach. Red circle are the detected local features, cyan lines are best match and white lines are second-best-match, matching procedure is analyzed over multiple frames

native Support Vector Machine [40][27][25] or bag-of-word codebook dictionary [10][36] to classify actions. As pointed out earlier, those methods only use the local feature information and neglect the global configuration constructed by the set of $\mathcal{AE}$. In our work, we aim to not only extract the rich local feature information but also the dynamic global structure of $\mathcal{AE}$. To this end, we select nonparametric Implicit Shape Model [29] as our learning approach, which is mainly based on a Voting Space implemented with Generalized Hough Transform [2].

## 4.1 Action Elements Analysis

For each set of Although $\mathcal{DAE}$ and $\mathcal{GAE}$ are detected in two different approaches, Figure 4.1 and 4.2, they both contain similar information about the motion and shape behavior of local salient regions. Using the representation notation from Section 3.1 $\mathcal{AE}(x, y, t, s, d, c, \omega)$, we have made the learning step transparent from feature extraction, that is, regardless of which feature extraction method we have carried out, as long as the extracted features are defined by $\mathcal{AE}(x, y, t, s, d, c, \omega)$, we will be able to run our learning procedure, this transparent development of feature-based is illustrated by a set of $\mathcal{AE}$ detected on KTH action *jogging* in Figure 4.3.

For each set of $\mathcal{AE}$ on one keyframe, we calculate a rectangular bounding box containing all these $\mathcal{AE}$ and apply a normalization process to align the centers of all these bounding boxes with the center point of the image, as shown on the last row of Figure 4.3. We denote the center of point of each normalized bounding box as Local Action Center $\mathcal{LAC}$, drawn as small yellow circles in the center position of white rectangle in Figure 4.3. For a list of all aligned $\mathcal{LAC}$ over time, we nominate the Global Action Center $\mathcal{GAC}$ as the mean point of all $\mathcal{LAC}$. With this normalized coordinate system, $x, y, t$ are now updated as the position of each $\mathcal{AE}$ relative to the $\mathcal{GAC}$ of each action. This normalization process is indispensable for our Implicit Shape Model approach, which is built upon geometric structure of all $\mathcal{AE}$.

Using feature description value $d$ from each $\mathcal{AE}$, we apply agglomerative clus-

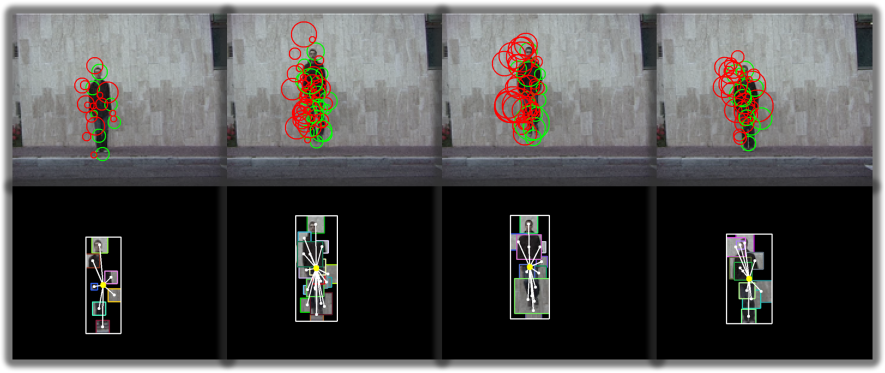


Figure 4.1: Detected $\mathcal{DAE}$ for Weizmann [18] action *pjump*

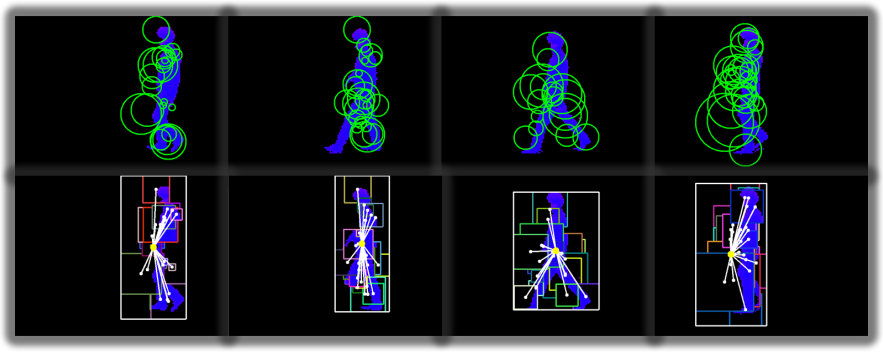


Figure 4.2: Detected $\mathcal{GAE}$ for KTH [40] action *walking*

tering technique used in [1] to group $\mathcal{AE}$ based on pair-wise Euclidean distance of two Action Element feature descriptors

$$E(d_i, d_j) = \sum_n (d_i(n) - d_j(n))^2 \tag{4.1}$$

In this notation, $n$ indicates descriptor element of feature vector $d$, in both cases of $\mathcal{DAE}$ and $\mathcal{GAE}$, $n$ indicates the gradient bin number. Agglomerative clustering in our case starts with each $\mathcal{AE}$ as a separate cluster, the cluster merge procedure will iteratively calculate the similarity distance $\phi(C_1, C_2)$ between two clusters ($C_1$ and $C_2$), they will only be merged if their similarity distance is higher than a threshold $\varphi$ to determine if they should be merged, as adopted from [1]

$$\phi(C_1, C_2) = \frac{\sum_{d_i \in C_1, d_j \in C_2} E(d_i, d_j)}{|C_1| \times |C_2|} > \varphi \tag{4.2}$$

This clustering scheme has its advantage over traditional $k - mean$ in the way that the clustering control factor is the similarity threshold $\varphi$ instead of $k$, number of clusters. Different video actions $\mathcal{A}$ will have many different dis-

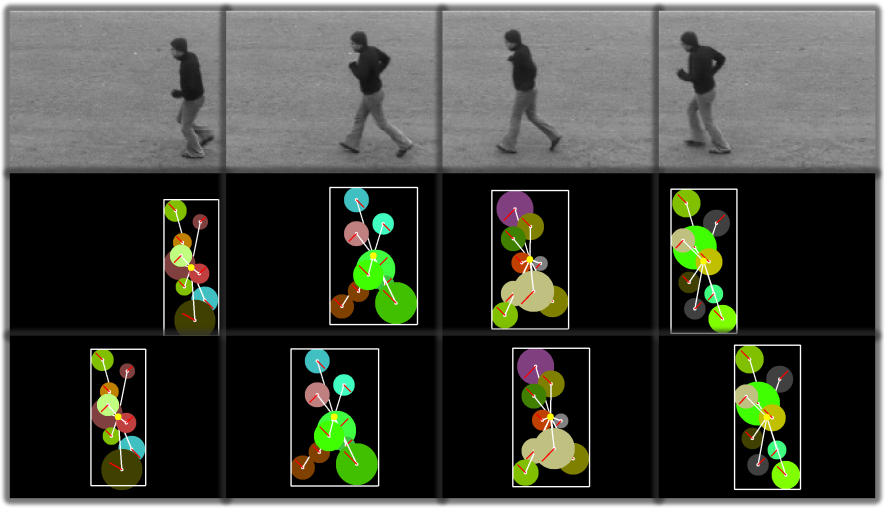


Figure 4.3: Detected $\mathcal{AE}$ for KTH action *jogging*. The learning process is made transparent with feature extraction approach, given standard representation $\mathcal{AE}(x, y, t, s, d, c, \omega)$

tribution of feature descriptors, hence the number of clusters will be varied in large interval. Meanwhile, in both $\mathcal{DAE}$ and $\mathcal{GAE}$ approaches, $d$ is the normalized histogram vector, which makes their Euclidean distance less sensitive to action type. In fact, using this clustering approach, we now have a fair and effective way to categorize $\mathcal{AE}$ into compact set of clusters. Each action $\mathcal{A}_u$ has its own cluster set $\mathcal{C}^u = \{< c, d >\}$ in which each $\mathcal{AE}$ will find its closest cluster identification $c$.

## 4.2 Hough Space Construction for ISM

Implicit Shape Model (ISM) as used for Object Recognition in [29] is a non-parametric voting procedure that incorporate votes into a common space which will be used later for inference. In our approach, we implement ISM of human action using Generalized Hough Transform [2]. Given an action $\mathcal{A}_u$, we can build a Hough Space $\mathcal{H}^u$ to contain all its extracted Action Elements $\mathcal{AE}^u$. $\mathcal{H}^u$ can be visualized as a Hash table illustrated in Figure 4.4. The Hash table $\mathcal{H}$ is represented as a set of all Action Element Classes $\mathcal{AEC}_{<\omega c>}$, specified by the index key pair $< \omega c >$. Action Element Class $\mathcal{AEC}$ is the term we use to describe a group of $\mathcal{AE}$, distinct by their cluster $c$ and gradient orientation $\omega$. In Figure 4.4, $\mathcal{AEC}$ are drawn as different monochrome columns starting from the front surface. By $\mathcal{AEC}$ definition, its size is determined by the product of cluster set size $\mathcal{C}$ and number of gradient orientations $\Omega$, $|\mathcal{AEC}| = |\mathcal{C}| \times |\Omega|$. Our experiments in Section 5.2 will show the average number of $\mathcal{AEC}$ detected using different Action Element Extraction. In our design, we choose four different gradient orientation for $\Omega$, as empirical experiments show that this value is sufficient, which makes the resulted Hough Space not too sparse (with many empty entries) and not too dense, the black circles with red lines on the left of

Hough Space in Figure 4.4 illustrates four dominant gradient orientations used in our model.

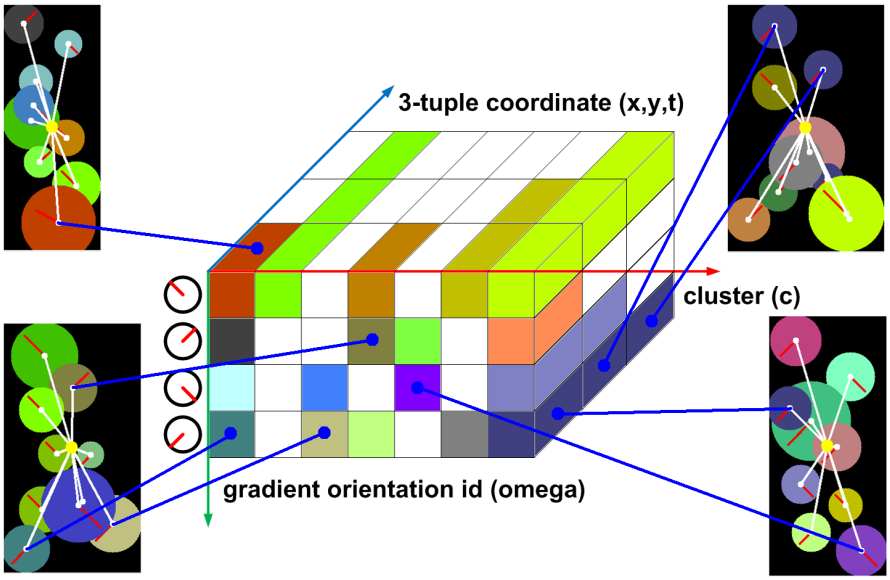



Figure 4.4: Filling Hough Space Procedure. Hough Space $\mathcal{H}$ is represented as the Hash table with index key pair $< \omega c >$ and location entries $(x, y, t)$. The four black circles on the left of $\mathcal{H}$ indicate 4 different gradient orientations according to the red line direction. The filling process goes through each $\mathcal{AE}_n$ to find its Action Element Class $\mathcal{AEC}_{<kl>}$ match, and fill accumulatively fill in its location content. $\mathcal{AEC}$ are represented as the monochrome columns generated from this filling process

The Hough Space $\mathcal{H}$ is constructed by filling all $\mathcal{AE}$ into the Hash table using the index key pair $< \omega c >$

$$\mathcal{H} = \{\mathcal{AEC}_{<kl>}\}, 1 \leq k \leq |\Omega|, 1 \leq k \leq |\mathcal{C}| \tag{4.3}$$

where each $\mathcal{AEC}_{<kl>}$ entry will be filled by all $\mathcal{AE}$ having the same cluster and gradient entities

$$\mathcal{AEC}_{<kl>} = \{\forall \mathcal{AE}_n | \omega_n = k, c_n = l\} \tag{4.4}$$

This filling process will run until all $\mathcal{AE}$ have been processed, and the Hough Space $\mathcal{H}$ can then be used for matching this model action to other action. The filled Hough Space conceptually contains all possible location votes for the Action Center $\mathcal{AC}$, organized trunks using index key of description cluster and gradient orientation.

## 4.3 Action Matching as Model Fitting

Using the Hough Space $\mathcal{H}^u$ constructed from action $\mathcal{A}_u$, the task of matching another action $\mathcal{A}_v$ with $\mathcal{A}_u$ now becomes the projection of $\mathcal{A}_v$ onto $\mathcal{A}_u$ Hough

Space $\mathcal{H}^u$, and the matching score is determined by the fitness of projected distribution. The projection of $\mathcal{A}_v$ onto $\mathcal{H}^u$ starts with defining the Projection Space $\mathcal{P}_v^u$ of $\mathcal{A}_v$ using $\mathcal{A}_u$ model. The size of $\mathcal{P}_v^u(w,h,l)$ is defined by $\mathcal{V}_v(w,h,l)$

$$\begin{aligned}\mathcal{P}_v(w) &= 2\mathcal{V}_v(w) - 1 \\ \mathcal{P}_v(h) &= 2\mathcal{V}_v(h) - 1 \\ \mathcal{P}_v(l) &= 2\mathcal{V}_v(l) - 1\end{aligned} \tag{4.5}$$

Each Action Element $\mathcal{AE}_m^v(x,y,t,s,d,c,\omega) \in \mathcal{AE}^v$ is used to find its closest match $\mathcal{AEC}_{kl}^u$ on Hough Space $\mathcal{H}^u$, and all Action Center vote casts from $\mathcal{AEC}_{kl}^u$ will be projected onto $\mathcal{P}_v^u$. The fourth row on Figure 4.5 illustrates the projecting procedure, with each colored circle Action Element $\mathcal{AE}_m^v$, there are several votes have been casted based on its best Action Element Class match obtain from $\mathcal{H}^u$.

After all $\mathcal{AE}^v$ has been used for projection, we now have a complete $\mathcal{P}_v^u$ containing all possible candidates for the Action Center $\mathcal{AC}_v^u$. Note here, we use $\mathcal{AC}_v^u$ to indicate the voted Action Center for action $\mathcal{A}_v$, which is different from $\mathcal{AC}^v$, the Action Center calculated directly from its Action Elements $\mathcal{AE}^v$.

This projection procedure can be formulated by defining a marginalization probability $p(\mathcal{AC}_v^u|\mathcal{AE}^v,\mathcal{L})$ to formulate the relationship of $p(\mathcal{AC}_v^u$ using location statistics $\mathcal{L} = \mathcal{AE}^v(x,y,t)$ and the voting index key pair $\eta = <\omega c>$ to find Action Center location $(x,y,t,$ this factorization is similar to the object ISM used in [29]

$$p(\mathcal{AC}_v^u|\mathcal{AE}^v,\mathcal{L}) = \sum_n p(\mathcal{AC}_v^u|\mathcal{AE}^v,\eta_n,\mathcal{L})p(\eta_n|\mathcal{AE}^v,\mathcal{L}) \tag{4.6}$$

$$= \sum_n p(\mathcal{AC}_v^u|\eta_n,\mathcal{L})p(\eta_n|\mathcal{AE}^v) \tag{4.7}$$

In our case, $p(\eta_n|\mathcal{AE}^v = \mathcal{AE}^v(\omega,c)$ is simply the two attributes of $(c,\omega)$ from detected $\mathcal{AE}^v$, and $p(\mathcal{AC}_v^u|\eta_n,\mathcal{L})$ is the retrieval of $\mathcal{AC}_v^u$ using index key $\eta_n$. The quantitative matching score of $\mathcal{A}_v$ and $\mathcal{A}_u$ can then be defined as the integral of all voted $\mathcal{AC}_v^u$

$$\Psi(\mathcal{A}_u,\mathcal{A}_v) = \sum_n \sum_m p(\mathcal{AC}_{vn}^u|\mathcal{AE}_m^v,\mathcal{L}_m) \tag{4.8}$$

In our work, we find the approximate solution for the matching function in Equation 4.8 using geometric characteristics of the Projection Space, observing that the matching score $\Psi(\mathcal{A}_u,\mathcal{A}_v)$ of two actions is proportional to the density distribution of the Projection Space $\mathcal{P}_v^u$. Similar observation has been drawn from [29] and they apply a density estimation approach using mean-shift Parzen window on two dimensional Projection Space. In our case, the Projection Space has three dimensions, so the complexity is extremely high and convergence is not guaranteed produce a fair estimation of the fitting model. Instead, we propose a much simpler but effective multi-dimensional density searching algorithm using approximate model weight.

We denote a Model Density Weight $\mathcal{W}$ to indicate the self-fitting density of an action model. That is, for an action $\mathcal{A}_u$, we construct the Project Space $\mathcal{P}_u^u$ by using the same action on its Hough Space $\mathcal{H}^u$. The resulted $\mathcal{P}_u^u$ will have

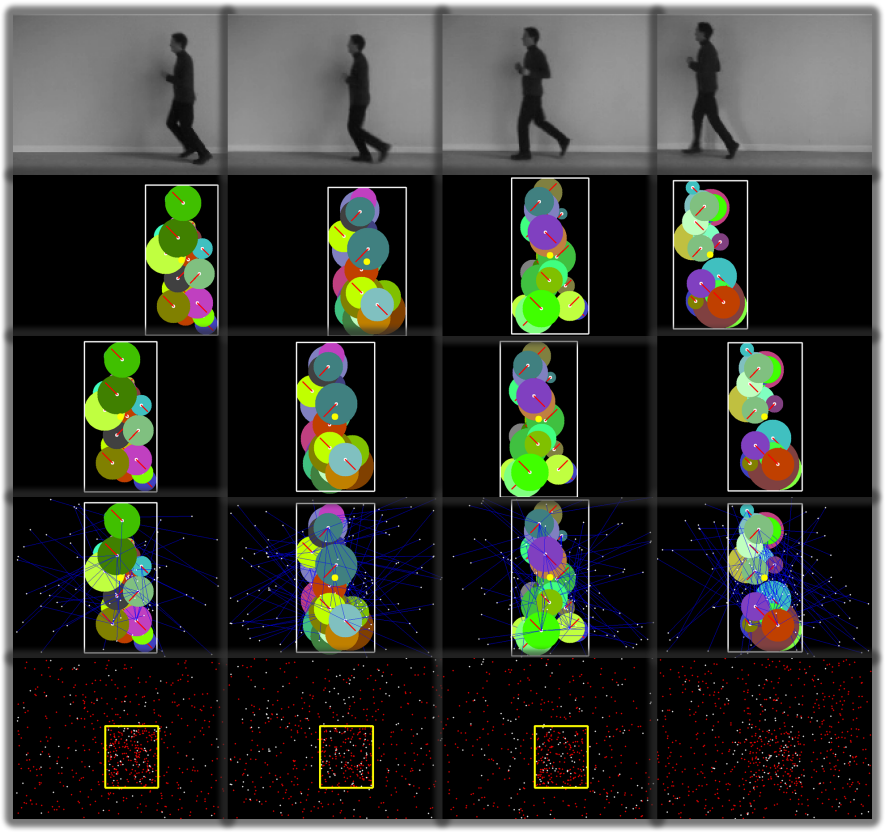


Figure 4.5: Matching Procedure for KTH [40] action $\mathcal{A}_v$ *jogging* using another *jogging* action model $\mathcal{A}_u$. 1. The first row shows 4 raw sample keyframes. 2. The second row shows the extracted Action Elements $\mathcal{AE}^v$. 3. Third row shows the repositioning using local center-wise. 4. The fourth rows illustrates the projection process using another *jogging* action model. The thin blue lines are the actual vote casts obtained from the Action Element Class $\mathcal{AEC}^u$ in model Hough Space $\mathcal{H}^u$, each $\mathcal{AE}^v$ might have more than one cast instances, the small white points at the end of blue lines are the voted Action Center $\mathcal{AC}_v^u$. Note that the actual Projection Space $\mathcal{P}_v^u$ is about two times larger in all dimensions, as given in Equation 4.5 but we only show these center regions. 5. The last row shows the complete votes on the $\mathcal{P}_v^u$, those white points are votes retrieved from $\mathcal{AE}^v$ in the same frame, while the red points come from $\mathcal{AE}^v$ of different time frame. The Model Fitting Region $\mathcal{MFR}_v^u$ are drawn in yellow rectangle containing the Model Density Weight $\mathcal{W}$ amount of votes, and will be used to calculate the Matching Score $\Psi(\mathcal{A}_u, \mathcal{A}_v)$

global maxima intensity $\Upsilon$ of the projected Action Center $\mathcal{AC}_u^u$ at the same location of model Action Center $\mathcal{AC}^u$. Model Density Weight $\mathcal{W}$ is defined as the ratio of this maxima intensity $\Upsilon$ (the amount of contributive votes) on the total number of projected points $|\mathcal{P}_u^u|$

$$\mathcal{W} = \frac{\Upsilon}{|\mathcal{P}_u^u|} \tag{4.9}$$

This $\mathcal{W}$ indicates how distinctive the extracted Action Elements are, and has a value close to, but not necessarily the same as $\frac{|\mathcal{AEC}|}{|\mathcal{AE}|}$, which is the number of Action Element Class on total number of Action Element. Using this value, which is representative for action model density, we define the Model Fitting Region $\mathcal{MFR}(w, h, l)$ inside the Projection Space $\mathcal{P}$ that contains the same percentage of $\mathcal{AC}$ votes as the Model Density Weight $\mathcal{W}$. Using the matching problem of $\mathcal{A}_v$ onto model $\mathcal{A}_u$, we now have

$$\mathcal{W} = \frac{|\mathcal{MFR}_v^u|}{|\mathcal{P}_v^u|} = \frac{\Upsilon}{|\mathcal{P}_u^u|} \tag{4.10}$$

This equation holds for the case of self-fitting where $\mathcal{MFR}_u^u$ is the model Action Center $\mathcal{AC}^u$ which makes $|\mathcal{MFR}_u^u| = \Upsilon$.

Using this Model Fitting Region $\mathcal{MFR}_v^u$, we now define the Fitting Score $\Re$ as the ratio of number of votes inside the $\mathcal{MFR}_v^u$ on the volume of $\mathcal{MFR}_v^u$, which is

$$\Re_v^u = \frac{|\mathcal{MFR}_v^u|}{\mathcal{MFR}_v^u(w) * \mathcal{MFR}_v^u(h) * \mathcal{MFR}_v^u(l)} * Z \tag{4.11}$$

with $Z$ is the normalization factor to make sure Equation 4.11 holds for the case of self-fitting. That is

$$\begin{aligned}
\Re_u^u &= \frac{|\mathcal{MFR}_u^u|}{\mathcal{MFR}_u^u(w) * \mathcal{MFR}_u^u(h) * \mathcal{MFR}_u^u(l)} * Z \\
&= \frac{\Upsilon}{1} * Z, \text{ for } \mathcal{MFR}_u^u \text{ is a point} && (4.12) \\
\therefore Z &= \frac{1}{\Upsilon}, \text{ for } \Re_u^u = 1, \text{ Self-Matching Score} && (4.13)
\end{aligned}$$

substitute $\Upsilon$ from Equation 4.10

$$\Upsilon = |\mathcal{MFR}_v^u| * \frac{|\mathcal{P}_u^u|}{|\mathcal{P}_v^u|} \tag{4.14}$$

and update $Z$ in Equation 4.13

$$Z = \frac{|\mathcal{P}_v^u|}{|\mathcal{MFR}_v^u| * |\mathcal{P}_u^u|} \tag{4.15}$$

finally, put $Z$ into Equation 4.11

$$\begin{aligned}
\Re_v^u &= \frac{|\mathcal{MFR}_v^u|}{\mathcal{MFR}_v^u(w) * \mathcal{MFR}_v^u(h) * \mathcal{MFR}_v^u(l)} * \frac{|\mathcal{P}_v^u|}{|\mathcal{MFR}_v^u| * |\mathcal{P}_u^u|} \\
&= \frac{|\mathcal{P}_v^u|}{\mathcal{MFR}_v^u(w) * \mathcal{MFR}_v^u(h) * \mathcal{MFR}_v^u(l) * |\mathcal{P}_u^u|} && (4.16) \\
&= \frac{|\mathcal{P}_v^u|}{\mathcal{MFR}_v^u(V) * |\mathcal{P}_u^u|}, \text{ with Volume notation } \mathcal{MFR}_v^u(V) && (4.17)
\end{aligned}$$

Equation 4.17 has completed our derivation of the Fitting Score value $\Re_v^u$ when projecting $\mathcal{A}_v$ onto $\mathcal{A}_u$ model space. Fitting Score $\Re_v^u$ can be solved geometrically, and as discussed earlier, this Fitting Score actually reflects on the Matching Score function in Equation 4.8

$$\Psi(\mathcal{A}_u, \mathcal{A}_v) \approx \Re_v^u = \frac{|\mathcal{P}_v^u|}{\mathcal{MFR}_v^u(V) * |\mathcal{P}_u^u|} \tag{4.18}$$

We now have a way to calculate the Matching Score of any two actions, using the Fitting Score developed upon Model Density Weight. Solving Equation 4.18 requires we to find the Model Fitting Region $\mathcal{MFR}_v^u$, which is known to lie within $\mathcal{P}_v^u$ and contain the same percentage of of $\mathcal{AC}_v^u$ votes as $\mathcal{W}$. We divide each dimension of $\mathcal{P}_v^u$ using corresponding bin numbers $B_w$, $B_h$, and $B_l$, and calculate the number of projected votes found in each bin. For each dimension, we find the peak bin $b*$ (as colored in black in Figure 4.6) and accumulatively adding adjacent bins until the total votes of selected bins reach $\mathcal{W}$. Repeating this searching procedure on three dimensions of $\mathcal{P}_v^u$, we will eventually have three rectangular areas created from selected bins on three dimensions, the intersection of these three planes is in fact the Model Fitting Region in three dimensions, which is used to calculate $\mathcal{MFR}_v^u(V)$ in Equation 4.17. Figure 4.6 visualizes our procedure, note here the number of bins selected on each dimension is not necessarily as displayed. In fact, the values for $B_w$, $B_h$, and $B_l$ should be selected carefully depending on the dimension of original video, in our approach we adaptively pick the number of bins based on the relative amount of motion detected compared to dimension of video shot.

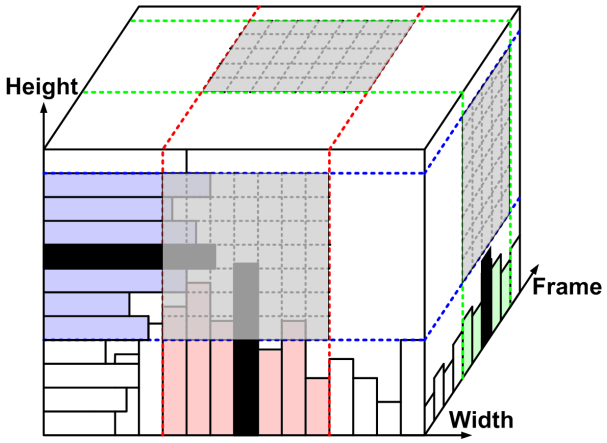



Figure 4.6: Geometric approach for finding Model Fitting Region $\mathcal{MFR}$ on Projection Space $\mathcal{P}$. Different bin numbers $B_w$, $B_h$, and $B_l$ are used to divide the dimension plane, the peak bin is colored in Black, its adjacent bins are collected so that the total percentage of votes reaches Model Density Weight $\mathcal{W}$. Intersections of these bins (grey rectangles) form the $\mathcal{MFR}$

The last row in Figure 4.5 draws the Model Fitting Region $\mathcal{MFR}$ in yellow rectangular for matching KTH action *jogging* onto Hough Space of another *jogging* action, while Figure 4.7 shows the case when the action *boxing* is matched

onto *jogging* model. We can roughly see that the matching case of same actions, the average amount of projected points $|\mathcal{P}_v^u|$ is higher and the Model Fitting Region region $\mathcal{MFR}_v^u(V)$ is smaller, which shows the Matching Score according to Equation 4.18 is higher than the case of testing different action class. Thorough experiments on performance of this Matching Score function will be presented in Section 5.3.

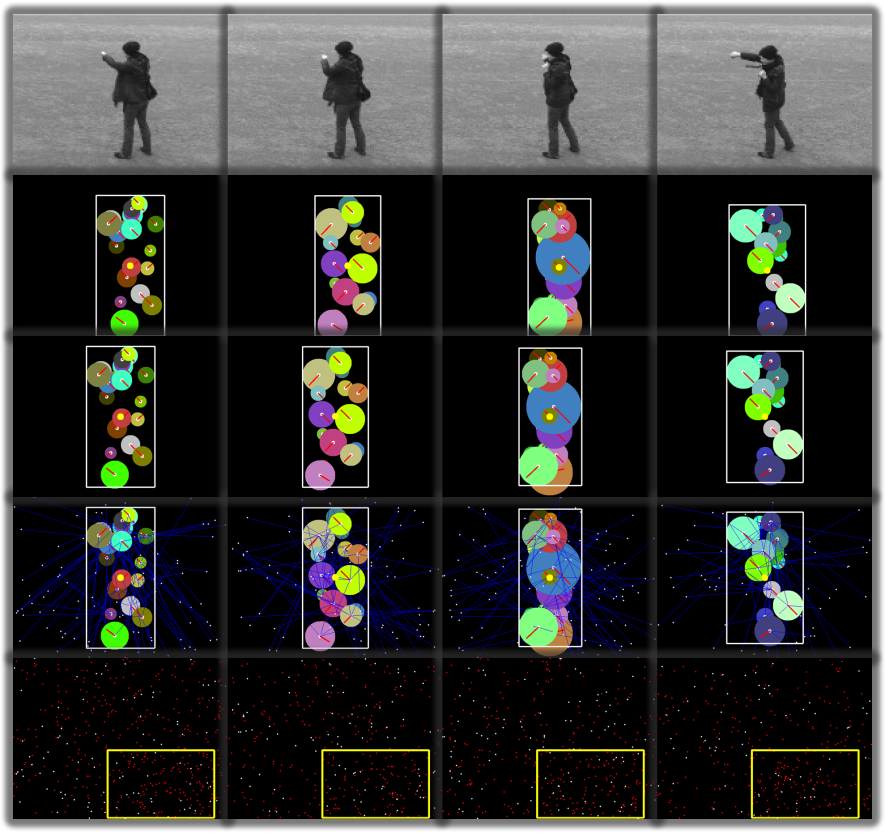


Figure 4.7: Matching Procedure for KTH [40] action $\mathcal{A}_v$ *boxing* using the same *jogging* action model $\mathcal{A}_u$ from Figure 4.5

We have presented our derivation of the Matching Score function $\Psi(\mathcal{A}_u, \mathcal{A}_v)$ and its geometric solution using intermediate notation of Model Fitting Region. It is noted that $\Psi(\mathcal{A}_u, \mathcal{A}_v)$ is a non-symmetrical function, that is matching score of $\mathcal{A}_v$ into $\mathcal{A}_u$) is not necessarily the same as matching $\mathcal{A}_u$ into $\mathcal{A}_v$), this behavior is typical for common Voting approaches in matching using pattern template. This matching score function $\Psi$ is useful for the case of fast action matching and retrieval using one-shot recognition, and in fact is used in our action retrieval task in Section 5.3 as the ranking criteria, higher matching score indicates higher matching certainty.

### 4.4 Action Distance Function for Action Classification

Using the matching score function, we develop a function for action distance $\mathcal{D}(\mathcal{A}_u, \mathcal{A}_v)$ as the logarithm average of reflective matching scores

$$\mathcal{D}(\mathcal{A}_u, \mathcal{A}_v) = \log \frac{\Psi(\mathcal{A}_u, \mathcal{A}_v) + \Psi(\mathcal{A}_v, \mathcal{A}_u)}{2} \tag{4.19}$$

Action Distance function $\mathcal{D}(\mathcal{A}_u, \mathcal{A}_v)$ is symmetric and represents the mutual distance between any two actions, also note that the distance of the two identical actions according to Equation 4.19 is zero. This Action Distance function is used in our action classification task, described in Section 5.4 to extract interdependent relationship of actions in the same classes and use them to recognize outliers of different action classes.

### 4.5 Action Localization

Apart from action retrieval and classification, our Implicit Shape Model approach can also help to do locate action regions in both space and time of the video source. Using the final Model Fitting Region $\mathcal{MFR}$, we backtrack on all Action Elements $\mathcal{AE}$ that contribute significant votes to this region, and select them as the localization segments for the detected action, as illustrated in Figure 4.8. In the two datasets KTH and Weizmann, there is no available information to carry out quantitative evaluation of the localization performance, so we only include in Figure 4.9 few qualitative snapshots of our segmentation task. Future works on different datasets can easily extend the current framework to do action localization.

## 5 Experimental Results

### 5.1 Dataset Selection and Experiment Setup

We use two fundamental action recognition datasets KTH [40] and Weizmann [18]. KTH has about 2400 greyscale video shots with 6 actions: *boxing*, *handwaving*, *handclapping*, *jogging*, *running*, *walking*, performed by 25 persons under 4 different contexts and subdivided into 4 intervals. Weizmann has about 90 colored video shots with 10 actions: *bend*, *jack*, *jump*, *pjump*, *run*, *side*, *skip*, *wave1*, *wave2*, *walk*, performed by 9 persons.

We first carry out analysis of the feature extraction task, analyzing performance of $\mathcal{DAE}$ and $\mathcal{GAE}$. We then evaluate the task of Action Retrieval using One-Shot training with Matching Score function $\Psi$. The Action Classification task is also carried out using fusion of information available from action instances in the same class with Action Distance function $\mathcal{D}$.

### 5.2 Action Elements Extraction

We run Action Element extraction using both Discriminative and Generative approaches on 16 action of the the datasets. We are particularly interested in the different number of total extracted Action Elements $|\mathcal{AE}|$ and categorized classes $|\mathcal{AEC}|$, according to our fixed selection in number of gradient orientations $\Omega$, $|\mathcal{AEC}|$ then directly reflects the number of descriptor clusters $|\mathcal{C}|$. Table 5.1

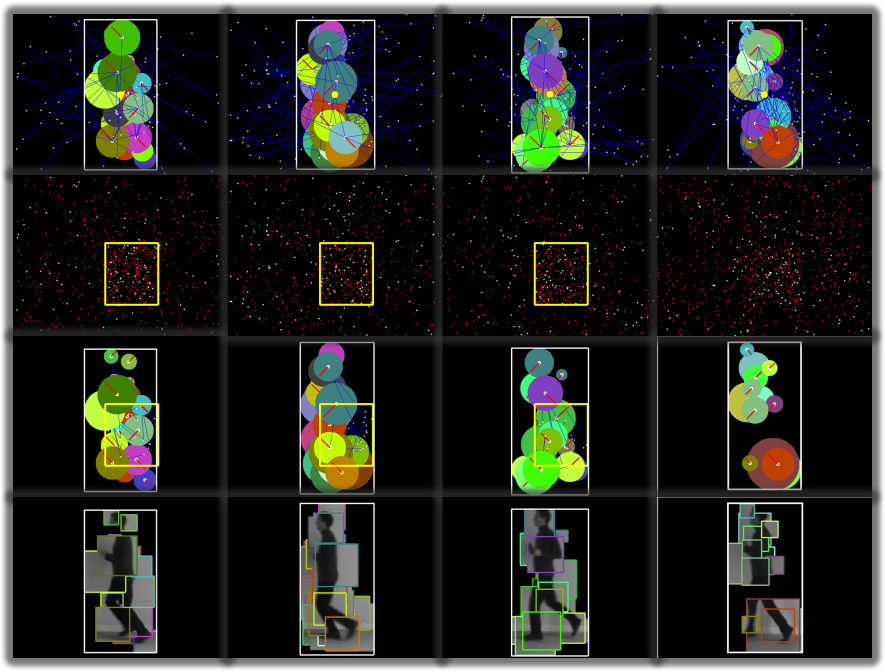


Figure 4.8: Action Localization. Using the Projection Space $\mathcal{P}$ and detected Model Fitting Region $\mathcal{MFR}$ in the first two rows, Action Localization is carried out as the backtracking on all Action Elements $\mathcal{AE}$ that contribute to the $\mathcal{MFR}$ These tracked $\mathcal{AE}$ are shown are masked and extracted from the original video as shown in the last row

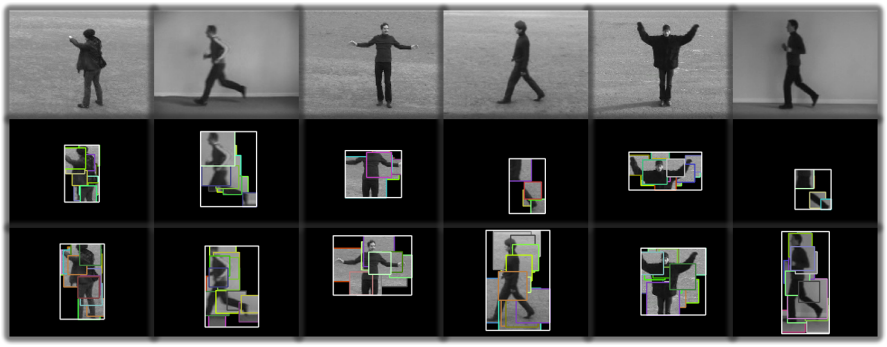


Figure 4.9: Action Localization on KTH, second row uses $\mathcal{DAE}$, third row uses $\mathcal{GAE}$

summarizes the sampled set of average $|\mathcal{AE}|$ and $|\mathcal{AEC}|$ categorized accordionist to actions and extraction approaches.

From Table 5.1, we can observe that generally $\mathcal{DAE}$ extracts less features than $\mathcal{GAE}$, this is reasonable since the MHI in $\mathcal{GAE}$ uses motion information varying in time. The number of elements per Action Element Class $\mathcal{AEC}$, which

| Action | $\mathcal{DAE}$ | | $\mathcal{GAE}$ | |
|---|---|---|---|---|
| | $\lvert\mathcal{DAE}\rvert$ | $\lvert\mathcal{DAEC}\rvert$ | $\lvert\mathcal{GAE}\rvert$ | $\lvert\mathcal{GAEC}\rvert$ |
| KTH *boxing* | 8143 | *472* | 9847 | 260 |
| KTH *handclapping* | **12754** | 1316 | **15379** | 684 |
| KTH *handwaving* | 10311 | 1244 | 11871 | 492 |
| KTH *jogging* | 4709 | 1868 | 7094 | 1272 |
| KTH *running* | 4993 | 1468 | 7850 | 1136 |
| KTH *walking* | 7209 | **2052** | 9561 | 1664 |
| Weizmann *bend* | 4956 | 528 | 6191 | *228* |
| Weizmann *jack* | 8495 | 668 | 9704 | 488 |
| Weizmann *jump* | 7276 | 636 | 7855 | 428 |
| Weizmann *pjump* | 4705 | 476 | 7009 | 388 |
| Weizmann *run* | 3943 | 540 | 4873 | 324 |
| Weizmann *side* | 5196 | 672 | 7933 | 568 |
| Weizmann *skip* | 3473 | 628 | 6246 | 564 |
| Weizmann *wave1* | *3389* | 536 | *3223* | 272 |
| Weizmann *wave2* | 4495 | 596 | 6401 | 540 |
| Weizmann *walk* | 7199 | 1068 | 8516 | 856 |

Table 5.1: Average Discriminative and Generative Action Elements for each action

is roughly seen as the ratio of $|\mathcal{DAE}|$ on $|\mathcal{DAEC}|$ is usually higher in $\mathcal{GAE}$, one of the reasons is the $\mathcal{GAE}$ approach works with binary data (encoded in MHI), so clustering feature distance is shorter than those in $\mathcal{DAE}$. We can also see that with action classes when the actors have much motion change, for instances *jogging*, *running*, *walking*, *side*, *skip*, the detection difference of two approaches in $|\mathcal{AE}|$ is much larger than other actions with less motion change, like in *boxing*, *bend*, *wave1*, and *wave2*.

We also collect the 8 most representative Action Elements for each type of action, shown in Figure 5.1 and 5.2. Generally, we can see that those extracted $\mathcal{AE}$ are mostly related to the human limbs, where most of the meaningful actions are carried out. For $\mathcal{GAE}$ with motion field feature, we can see the most common elements are those at the edges of each action region.


Figure 5.1: Representative Discriminative Action Elements $\mathcal{DAE}$


Figure 5.2: Representative Generative Action Elements $\mathcal{GAE}$

## 5.3 Action Matching and Retrieval

For the task of Action Retrieval on these two datasets, we use the proposed Matching Score function $\Psi(\mathcal{A}_u, \mathcal{A}_v)$ described in Section 4.3. We carry out OneClip action retrieval, select one video action $\mathcal{A}_u$ at a time to build the action model, and use Matching Score function to $\Psi$ evaluate the similarity of video shots from the testing pool. Figure 5.3 shows a snapshot of our action retrieval system, implemented as visual-based video search engine. Ranking the matching score in ascending order, we then have a way to quantitatively evaluate the retrieval performance, the Receiver Operating Characteristic (ROC) in Figure 5.4 shows the performance using Implicit Shape Modeling of two kinds of Action Elements.

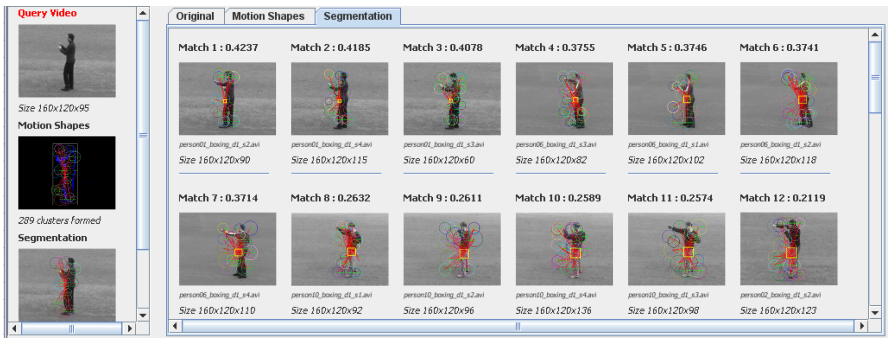



Figure 5.3: Graphical User Interface of our implemented visual-based video search engine. Left Panel shows the action model $\mathcal{A}_u$ KTH *boxing*. Right Panel displays the best matches returned, ranked using ratio Matching Score function to $\Psi$

The straight black line in Figure 5.4 is the Cut-off line, which connects the two ends of True Positive Rate [0,1] and False Positive Rate [1,0]. Intersections of this line with ROC curves are called Cut-off points (drawn in black circles), indicating the position where Sensitivity is equal to Specificity, and normally used to analyze the average performance. At these intersect, we recalculate the total true positive and false positive according to actions and plot 4 corresponding Confusion matrix with in Figure 5.5. Figure 5.4 and 5.5 show that

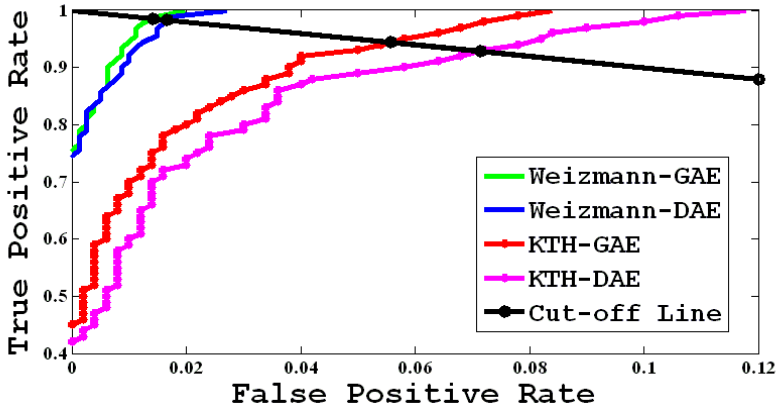
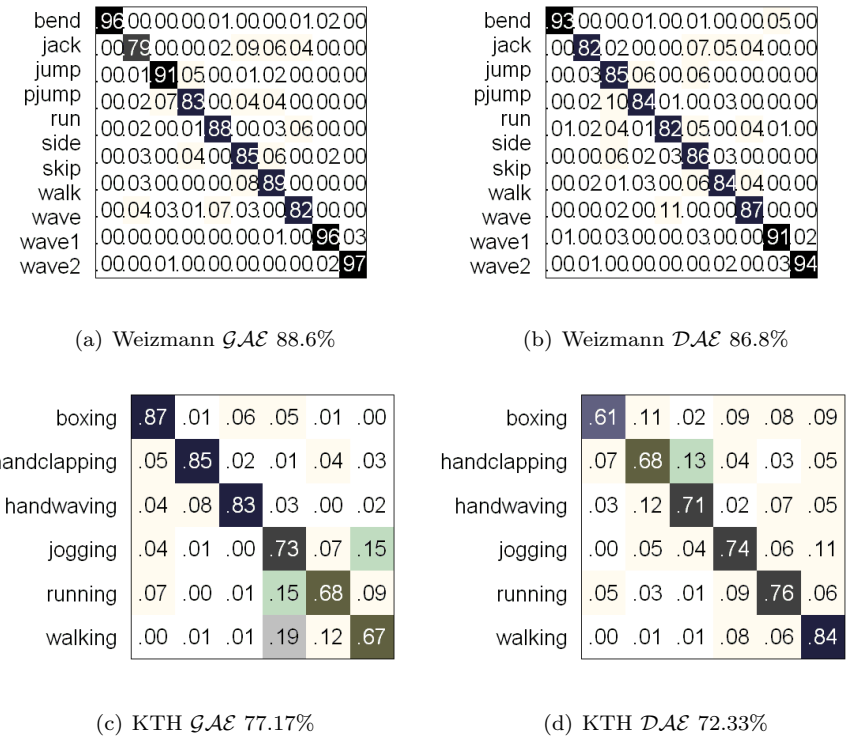



Figure 5.4: ROC for OneClip Action Retrieval




(a) Weizmann $\mathcal{GAE}$ 88.6%

(b) Weizmann $\mathcal{DAE}$ 86.8%

(c) KTH $\mathcal{GAE}$ 77.17%

(d) KTH $\mathcal{DAE}$ 72.33%

Figure 5.5: Confusion Matrices for One-Shot Action Retrieval Task

generally experiments on Weizmann yield better result than on KTH, which is reasonable provided that the background in Weizmann is static, and its actions stable and distinctive. It is also noted that for One-Shot Action Retrieval, $\mathcal{GAE}$ outperforms $\mathcal{DAE}$ on both datasets, where in KTH the performance difference is somewhat higher than in Weizmann. One of the possible explanation might

be $\mathcal{GAE}$ includes more Action Elements into learning procedure, and for this particular retrieval case, there is only one given video shot as training, so $\mathcal{DAE}$ hardly generates enough discriminative inference on the action. For action classification results, generally periodic actions (*bend*, *wave*, *boxing*) have higher performance than non-periodic actions (*running*, *walking*, *jack*). Surprisingly, the two simple actions of boxing and handclapping are not detected well using $\mathcal{DAE}$, the possible reason is because some of the salient parts are not included in the model. Meanwhile, $\mathcal{GAE}$ severely misclassifies *jogging*, *running* and *walking* on KTH, their similar motion field might have caused this.

## 5.4 Action Classification

In order to produce a fair comparison with state-of-the-art in the field, we extend our action matching algorithm to deal with classification using multiple trainings. The common setup for classification task is 2/3 Split on KTH, that is, 16 persons are used for training and 9 others for testing. In our experiments, we select one video shot per context, person and action from 16 persons, which together of 384 video shots, and run test on 9 other persons, the same amount as common reported works. On Weizmann, we use the usual scheme as other works, which is Leave-One-Out, use 8 persons training at a time and iteratively run classification test until all video shots are used for training and testing.

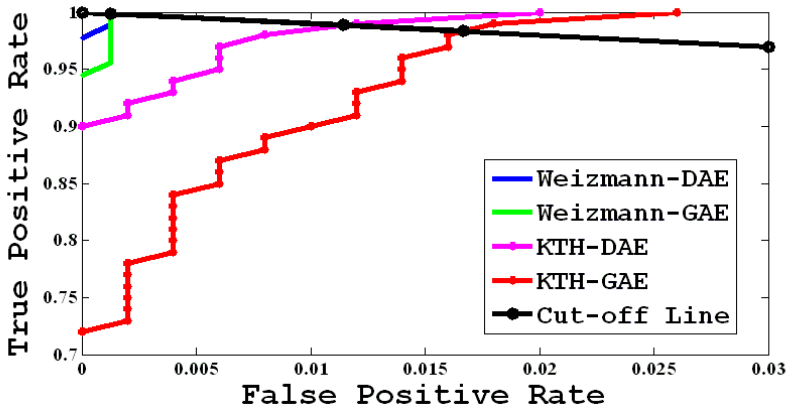
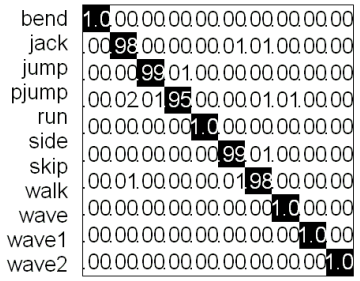



Figure 5.6: ROC for multiple training Action Classification

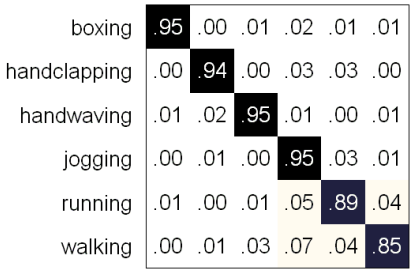
For the task of action classification as described in Section 4.4, we Action Distance function $\mathcal{D}$ to run on instances of the same class to find the distance interval that are distinctive and contribute to the classification of one class from the others. This mutual information between action instances of the same classes are integrated with the external matching score of actions outside the class. Using these two values, we have a better way to validate on the classification results and to be able to choose the closest model to performance classification. Superficially for the task of $\mathcal{DAE}$, we include more training into the SBFT to boost up the feature selection performance. Figure 5.6 shows the ROC curve for our classification performance. Similar usage with Cut-off line as explained

from previous section, we analyze the intersection to generate 4 corresponding Confusion matrices in Figure 5.7

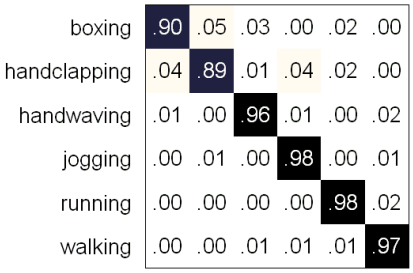



(a) Weizmann GAE 98.9%


(b) Weizmann DAE 98.9%




(c) KTH GAE 92.17%




(d) KTH DAE 94.67%

Figure 5.7: Action Classification Results

The overall performance is boosted using training fusion of multiple action instances. Surprisingly, $\mathcal{DAE}$ surpasses $\mathcal{GAE}$ on KTH and performs slightly better in Weizmann, as shown in the ROC. The main reason might be due to the fact that we have used more data to train the Sparse Bayesian Feature Classifier in $\mathcal{DAE}$, that helps to extract more salient Action Elements for action learning. We again observe that $\mathcal{GAE}$ generally works better in the case of small distinctive actions like boxing, handclapping, while in the actions where large motion field is included like running, walking, they increase number of redundant local features in $\mathcal{GAE}$ and eventually when it comes to Implicit Shape Model learning, those noisy features increase the error bound of the voting space and reduce greatly the overall performance. It is also learned from these two set of experiments that, while Generative model is normally good for the case of little training is provided, like in OneClip action retrieval or video search, the Discriminative model adapts quickly to rich training data, and be able to draw distinctive inference on the mutual relationship between instances in and outside an action class.

Using the average classification performance from these two tasks, we conduct a comparison survey with the most recent state-of-the-art action recognition works, as shown in Table 5.2, ordering based on performance. Interestingly, our methods slightly outperform [52] in the context of OneClip classification.

| **Author** | **KTH** | **Weizmann** |
|---|---|---|
| $\mathcal{DAE}$ | ***94.67*** | ***98.9*** |
| $\mathcal{GAE}$ | ***92.17*** | ***98.9*** |
| $\mathcal{GAE}$ **OneClip** | ***77.17*** | ***88.6*** |
| $\mathcal{DAE}$ **OneClip** | ***72.33*** | ***86.8*** |
| Weinland and Boyer[53] | * | 100 |
| Gorelick et al.[18] | * | 99.6 |
| Liu and Shah[30] | 94.2 | * |
| Sun and Hauptmann[46] | 94 | 97.8 |
| Grundmann et al.[19] | 93.52 | 96.39 |
| Mikolajczyk and Uemura[33] | 93.2 | * |
| Schindler and Van Gool[39] | 92.7 | 100 |
| Laptev et al.[26] | 91.80 | * |
| Jhuang et al.[21] | 91.70 | 98.8 |
| Wang and Mori[51] | 91.17 | 98.33 |
| Fathi and Mori[12] | 90.50 | 100 |
| Rapantzikos et al.[38] | 88.30 | * |
| Jiang et al.[22] | 84.40 | * |
| Willems et al.[55] | 84.36 | * |
| Niebles et al.[36] | 81.50 | 72.8 |
| Dollar et al.[10] | 81.20 | * |
| Ke et al.[23] | 80.90 | * |
| Weilong Yang and Mori[52] Tr OneClip | 75.71 | * |
| Weilong Yang and Mori[52] Dc OneClip | 72.48 | 87 |
| Schuldt et al.[40] | 71.70 | * |

Table 5.2: Classification Performance of our work and state-of-the-arts

Compared to others, our proposed approach performs equally well on both dataset, and with sufficiently trained model using $\mathcal{DAE}$, we achieve the best classification score on KTH.

# 6 Conclusion

In this paper we presented a new solution for the challenge of action matching, retrieval and classification in video. By decomposing a video action into linked set of sparse Action Elements, we develop two feature extraction approaches based on recent generative and discriminative techniques to extract the most compact Action Elements from the video. We also use an Implicit Motion Shape Model, implemented as a Generalized Hough Space, to intergrade the global structure of all Action Elements into one dynamic structured model. We derive an action matching and an action distance function for video actions. Matching is done based on the density estimation of Model Fitting Region, solved using geometric characteristic of the model. Outstanding results on two common benchmarks of action recognition KTH and Weizmann have proved the effectiveness and robustness of our proposed framework.


# Acknowledgement

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and Digital Economy and the Australian Research council through the ICT Centre of Excellence program.


# A A Self-Fitting Case for Model Density Weight

In this section, we will consider the convergence likelihood of the voting space for motion-shape structure. We use a case study of a fundamental $3x3$ square having 8 local patches extracted around its center, as shown in Figure

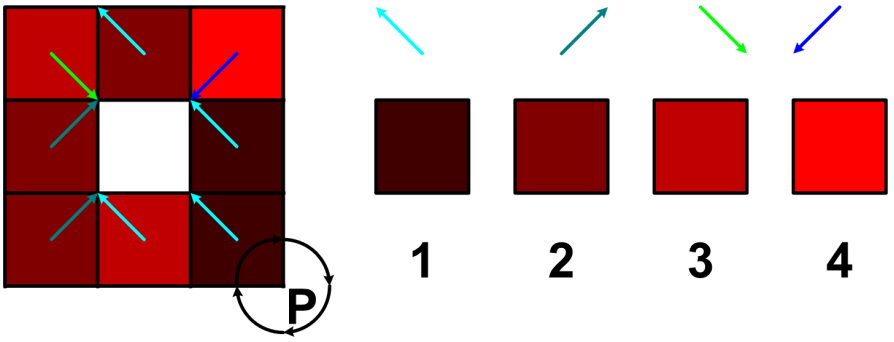
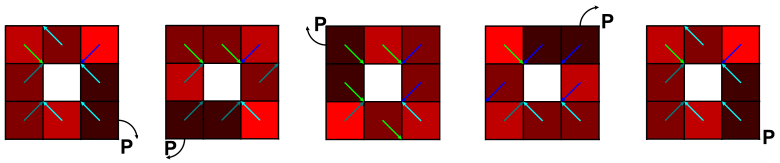



Figure A.1: Fundamental

In this case, all the frames have the same dimension so we only need to relocate them such that their center pixels are in the middle. We easily realize that the nominated Action Center is actually the center pixel of Frame 3 (in




(a) Frame 1 (b) Frame 2 (c) Frame 3 (d) Frame 4 (e) Frame 5

Figure A.2: Aligned Matching Results based on Action

| | $c_1$ | $c_3$ | $c_2$ | $c_2$ | $c_3$ | $c_2$ | $c_4$ | $c_1$ |
|---|---|---|---|---|---|---|---|---|
| Frame 1 | Ω1 | Ω1 | Ω2 | Ω2 | Ω3 | Ω1 | Ω4 | Ω1 |
| Frame 2 | Ω2 | Ω2 | Ω3 | Ω3 | Ω4 | Ω2 | Ω1 | Ω2 |
| Frame 3 | Ω3 | Ω3 | Ω4 | Ω4 | Ω1 | Ω3 | Ω2 | Ω3 |
| Frame 4 | Ω4 | Ω4 | Ω1 | Ω1 | Ω2 | Ω4 | Ω3 | Ω4 |
| Frame 5 | Ω1 | Ω1 | Ω2 | Ω2 | Ω3 | Ω1 | Ω4 | Ω1 |

Table A.1: Cluster Ω data

Figure A.4). Using this Action Center coordinate system, we can then calculate the 3-tuple $(x, y, t)$ relative position of every Action Element and construct the Hough Space based on this accumulation. The resulted Hough Space of this square rotation action is shown in Table A.2

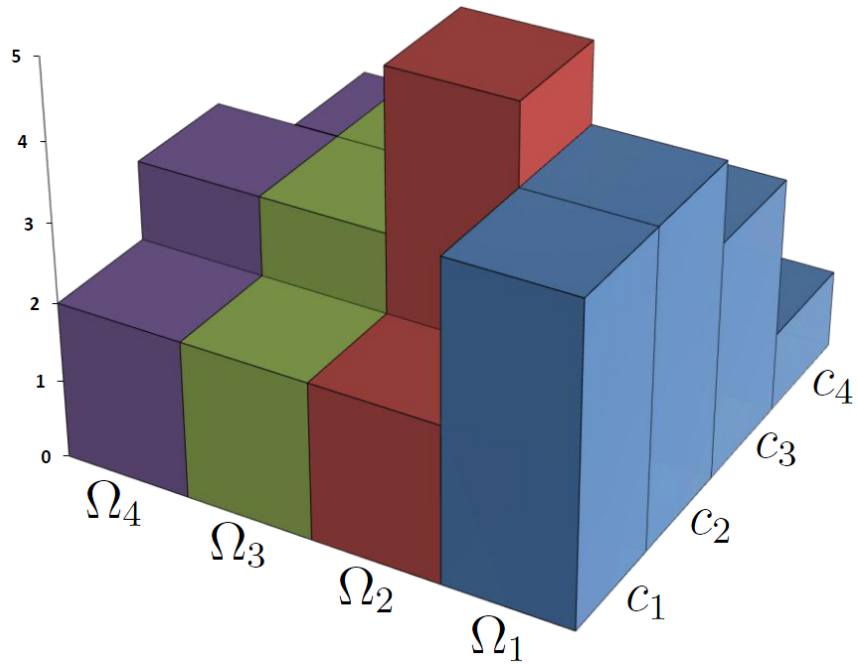



Figure A.3: Graphical Hough Space

| $(x, y, t)$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ |
|---|---|---|---|---|
| $\Omega_1$ | (-1,-1,2)<br>(-1,0,2)<br>(-1,-1,-2)<br>(-1,0,-2) | (-1,-1,-1)<br>(0,-1,-1)<br>(0,1,2)<br>(0,1,-2) | (0,-1,2)<br>(0,-1,-2)<br>(-1,-1,0) | (-1,-1,1) |
| $\Omega_2$ | (1,-1,1)<br>(0,-1,1) | (1,-1,2)<br>(1,0,2)<br>(1,-1,-2)<br>(1,0,-2)<br>(-1,0,1) | (1,0,1)<br>(1,-1,-1) | (1,-1,0) |
| $\Omega_3$ | (1,1,0)<br>(1,0,0) | (1,1,1)<br>(0,1,1)<br>(0,-1,0) | (0,1,0)<br>(1,1,2)<br>(1,1,-2) | (1,1,-1) |
| $\Omega_4$ | (0,1,-1)<br>(-1,1,-1) | (-1,1,0)<br>(-1,0,0)<br>(1,0,-1) | (-1,0,-1)<br>(-1,1,1) | (-1,1,2)<br>(-1,1,-2) |

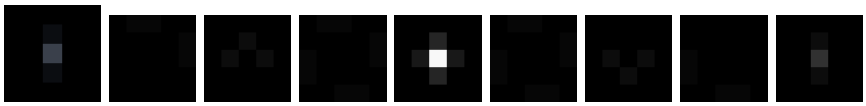
Table A.2: Voting Entries for Hough Space

In this model, there are 40 Action Elements grouped into 16 categories using 4 different $\Omega$ and $c$ values. The convergence test of voting space is carried out by self-fitting the same motion sequence to the generated model. The possible projection space consists of $l_p = 9$ planes, each with dimension $7 \times 7$ calculated from $l_v = 5$, $w_v = 3$, and $h_v = 3$ according to Equation 4.5.

| Plane 1 | | | | | Plane 2 | | | | | Plane 3 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |
| 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 2 | 0 |
| 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Plane 4 | | | | | Plane 5 | | | | | Plane 6 | | | | |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 4 | **40** | 4 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| Plane 7 | | | | | Plane 8 | | | | | Plane 9 | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 |
| 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 |
| 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Table A.3: Complete Projected Planes of Self-Fitting Case

In this case, the Model Fitting Region (MFR) is the center point of the fifth projected plane, with Model Density Ratio $\Re = 0.3$ according to Equation



(a) Pl_1 (b) Pl_2 (c) Pl_3 (d) Pl_4 (e) Pl_5 (f) Pl_6 (g) Pl_7 (h) Pl_8 (i) Pl_9

Figure A.4: Projected Distribution

4.16. We can see that this is a high value since the motion model of this object is simple and the detected Action Elements are more generic with average of 2.5 Action Elements per category. With more complex structures and motion behaviors, $\Re$ will be much less, as analyzed in the next Appendix.

# B Snapshots for Action Elements Extraction

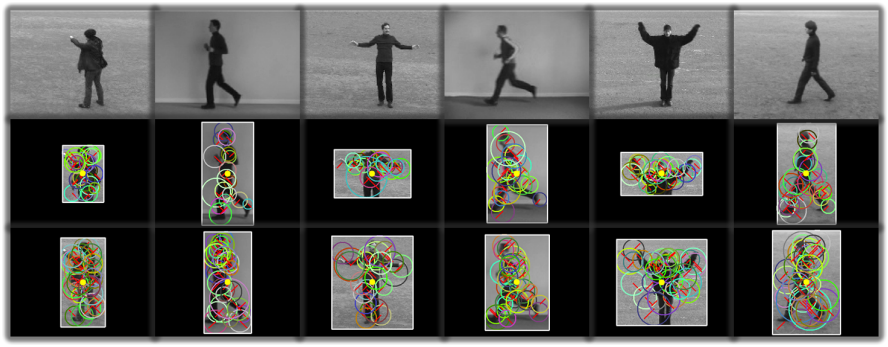


Figure B.1: Snapshots of detected Action Elements on KTH, first row is the raw video, second row is $\mathcal{DAE}$, last row is $\mathcal{GAE}$

Figure B.2: Snapshots of detected Action Elements on Weizmann, first row is the raw video, second row is $\mathcal{DAE}$, last row is $\mathcal{GAE}$

# Bibliography


[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. *Lecture Notes in Computer Science*, pages 113–130, 2002.

[2] DH Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[3] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.

[4] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. on Pattern Anal. and Machine Intell*, 2001.

[5] P. Carbonetto, G. Dorkó, C. Schmid, H. Kuck, and N. De Freitas. A semi-supervised learning approach to object recognition with spatial integration of local features and segmentation cues. *LECTURE NOTES IN COMPUTER SCIENCE*, 4170:277, 2006.

[6] P. Carbonetto, G. Dorkó, C. Schmid, H. K
"uck, and N. de Freitas. Learning to recognize objects with little supervision. *International Journal of Computer Vision*, 77(1):219–237, 2008.

[7] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proc. ICCV*, volume 2. Citeseer, 2007.

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, volume 1, page 22. Citeseer, 2004.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 886. Citeseer, 2005.

[10] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. *ICCV VS-PETS*, 2005.

[11] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *IEEE International Conference on Computer Vision*, volume 2, pages 726–733. Citeseer, 2003.

[12] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *IEEE Computer Society Conference on Computer Vision and Pattern Recongition*, 2008.

[13] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. Citeseer, 2005.

[14] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. Citeseer, 2008.

[15] R. Fergus, P. Perona, A. Zisserman, et al. Object class recognition by unsupervised scale-invariant learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. Citeseer, 2003.

[16] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminant models for object category detection. In *Proceedings of the 10th IEEE International Conference on Computer Vision, Beijing, China*. Citeseer, 2005.

[17] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globallyconsistent local distance functions for shape-based image retrieval and classification. In *Proc. ICCV*, volume 2. Citeseer, 2007.

[18] L. Gorelick, M. Galun, E. Sharon, R. Basri, and A. Brandt. Shape representation and classification using the poisson equation. In *IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*, volume 2. IEEE Computer Society; 1999, 2004.

[19] M. Grundmann, F. Meier, and I. Essa. 3D shape context and distance transform for action recognition. In *Proceedings of the International Conference on Pattern Recognition*, volume 1, pages 1–4, 2008.

[20] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.

[21] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *Proc. IEEE Int. Conf. Computer Vision*, volume 1, page 5. Citeseer, 2007.

[22] H. Jiang, M.S. Drew, and Z.N. Li. Successive convex matching for action detection. In *IEEE CVPR*, 2006.

[23] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *IEEE International Conference on Computer Vision*, volume 23, pages 38–41. Citeseer, 2007.

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

[25] I. Laptev, B. Caputo, C. Sch
"uldt, and T. Lindeberg. Local velocity-adapted motion events for spatio-temporal recognition. *Computer Vision and Image Understanding*, 108(3):207–229, 2007.

[26] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. *Proc. Conf. Computer Vision and Pattern Recognition*, 1:20–23, 2008.

[27] I. Laptev and P. Pérez. Retrieving actions in movies. In *International Conference on Computer Vision, Rio de Janeiro, Brazil*. Citeseer, 2007.

[28] L. Lee and WEL Grimson. Gait analysis for recognition and classification. In *Proceedings of the IEEE Conference on Face and Gesture Recognition*, pages 155–161, 2002.

[29] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV04 workshop on statistical learning in computer vision*, pages 17–32. Citeseer, 2004.

[30] J. Liu and M. Shah. Learning human actions via information maximization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recongition*, 2008.

[31] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[32] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[33] K. Mikolajczyk and H. Uemura. Action recognition with motion-appearance vocabulary forest. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[34] P. Moreels and P. Perona. A Probabilistic Cascade of Detectors for Individual Object Recognition. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, page 439. Springer, 2008.

[35] Y. Mu, S. Yan, T. Huang, and B. Zhou. Contextual motion field-based distance for video analysis. *The Visual Computer*, 24(7):595–603, 2008.

[36] J.C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

[37] M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1–8. Citeseer, 2007.

[38] K. Rapantzikos, Y. Avrithis, and S. Kollias. Dense saliency-based spatiotemporal feature points for action recognition. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[39] K. Schindler and L. Van Gool. Action snippets: How many frames does human action recognition require. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[40] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, 2004.

[41] J. Shotton, M. Johnson, R. Cipolla, T.C.R.D. Center, and J. Kawasaki. Semantic texton forests for image categorization and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[42] J. Sivic, B.C. Russell, A. Efros, A. Zisserman, and W.T. Freeman. Discovering object categories in image collections. In *Proc. ICCV*, volume 2, 2005.

[43] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, volume 2, pages 1470–1477. Citeseer, 2003.

[44] N. Snavely, S.M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *International Journal of Computer Vision*, 80(2):189–210, 2008.

[45] E.B. Sudderth, A. Torralba, W.T. Freeman, and A.S. Willsky. Describing visual scenes using transformed objects and parts. *International Journal of Computer Vision*, 77(1-3):330, 2008.

[46] X. Sun and M.C.A. Hauptmann. Action recognition via local descriptors and holistic features. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, 2009.

[47] Tuan Hue Thi, Sijun Lu, Jian Zhang, Li Cheng, and Li Wang. Human body articulation for action recognition in video sequences. *Advanced Video and Signal Based Surveillance, IEEE Conference on*, 0:92–97, 2009.

[48] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.

[49] R. Venkatesh Babu and KR Ramakrishnan. Recognition of human actions using motion history information extracted from the compressed video. *Image and Vision Computing*, 22(8):597–607, 2004.

[50] Y. Wang and G. Mori. Learning a discriminative hidden part model for human action recognition. *Advances in Neural Information Processing Systems (NIPS)*, 21:1721–1728, 2008.

[51] Y. Wang and G. Mori. Human action recognition by semi-latent topic models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1762–1774, 2009.

[52] Yang Wang Weilong Yang and Greg Mori. Human Action Recognition from a Single Clip per Action. In *2nd International Workshop on Machine Learning for Vision-based Motion Analysis (at ICCV)*, pages 1–8, 2009.

[53] D. Weinland and E. Boyer. Action recognition using exemplar-based embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 13, 2008.

[54] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006.

[55] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *Proc. European Conf. Computer Vision*, 23:650–653, 2008.

[56] H. Zhang, A.C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proc. CVPR*, volume 2, pages 2126–2136. Citeseer, 2006.