Vis Comput (2011) 27:1115–1123
DOI 10.1007/s00371-011-0656-x

ORIGINAL ARTICLE

Simultaneous tracking and action recognition for single actor human actions

Vivek Kumar Singh · Ram Nevatia

Published online: 6 November 2011

Abstract This paper presents an approach to simultaneously tracking the pose and recognizing human actions in a video. This is achieved by combining a Dynamic Bayesian Action Network (DBAN) with 2D body part models. Existing DBAN implementation relies on fairly weak observation features, which affects the recognition accuracy. In this work, we use a 2D body part model for accurate pose alignment, which in turn improves both pose estimate and action recognition accuracy. To compensate for the additional time required for alignment, we use an action entropy-based scheme to determine the minimum number of states to be maintained in each frame while avoiding sample impoverishment. In addition, we also present an approach to automation of the keypose selection task for learning 3D action models from a few annotations. We demonstrate our approach on a hand gesture dataset with 500 action sequences, and we show that compared to DBAN our algorithm achieves 6% improvement in accuracy.

Keywords Human action recognition · Dynamic Bayesian network · Pictorial structure

1 Introduction

The objective of this work is to recognize single actor human actions in videos captured from a single camera. Automatic human action recognition has a wide range of applications including human-computer interaction (HCI), visual surveillance and automatic video retrieval and has been a topic of active research in computer vision. Existing approaches differ on how the actions are modeled and how they are matched to the observations. In this work, we represent the actions as a sequence of simple action primitives represented in a Dynamic Bayesian Network (DBN), referred to as a Dynamic Bayesian Action Network (DBAN). Most likely activity sequences of actions, based on observations are computed from the DBAN. Observations are derived from the shape of the extracted foreground blobs corresponding to human actors. Our work closely follows the approach described in [13], but that work uses only the overall shape of the blobs as descriptors whereas we incorporate a more elaborate part-based analysis and show resulting improvements in performance. To distinguish our method from that of [13], we refer to the method used in [13] as DBAN-FGM (FGM standing for foreground matching)

In [16], we proposed an extension to DBAN, *DBAN-Parts* that use an intermediate 2D body part representation of the human model to accurately match the human model and image observations across shape variations and observation noise. Given a person scale and approximate viewpoint, the 3D pose is orthographically projected to 2D to determine the visible parts. A 2D part-based model [3] is then used to accurately align the 2D pose. The likelihood of the pose to recognize human actions is then computed over the aligned parts. This allows for more accurate local search for matching the 3D model to the image observation, resulting in more accurate action recognition. Furthermore, fewer samples need to be maintained which partially compensate for the time required to compute pose alignment. To further speed up the inference, we automatically determine the minimal number of hypotheses to be maintained at each step by defining an action class entropy. This paper extends our pre-

V.K. Singh (✉) · R. Nevatia
University of Southern California, Los Angeles, CA 90089, USA
e-mail: vivekks@gmail.com

R. Nevatia
e-mail: nevatia@usc.edu

vious conference publication [16].[1] Here, we also present a novel approach to automatically obtain primitive boundaries instead of manually selecting the boundaries (by selecting keyposes) as in [13, 16].

While [13] uses DBAN with foreground-based features for matching poses and shows good results on datasets with large pose variations such as Grocery Store and Weizmann set [2, 13], the recognition accuracy on the Gesture dataset with subtle pose variations is quite low, especially with fewer training data; the Gesture dataset [13] has about 500 segmented action sequences with a variety of arm gestures common in HCI applications. In this work, we show results on the Gesture dataset and demonstrate that using the 2D part model to compute the pose likelihood allows for a more accurate action recognition and pose estimation.

In the rest of the paper, we first review the related work in Sect. 2. We then discuss action model representation in Sect. 3; we also briefly describe how the actions are mapped to a Dynamic Bayesian Action Network. Next, we present a modified inference algorithm for efficient pose tracking and action recognition over DBAN in Sect. 4, followed by the part model representation and alignment in Sect. 5, results in Sect. 6 and conclusion in Sect. 7.

2 Related work

A popular approach to recognizing human actions is to use histograms of sparse spatio-temporal features [7] and use a classifier (such as SVM) to determine the action label. Such approaches are attractive in that because no explicit action modeling is required but they require a large amount of training data to capture viewpoint and other variations; they are also difficult to apply to the task of continuous action recognition. An alternative approach is to use graphical models to represent the evolution of the actor state in a video, for e.g. using HMMs [14], DBNs [10, 13] and CRFs [11, 12, 17]. The actor state is generally represented using a human model with 3D joint positions [10], 2D part templates [5] or an implicit representation using latent variables [4, 19]. Learning these models requires Motion Capture (MoCAP) data which can be difficult to collect.

Recently, [13] proposed a method to learn Dynamic Bayesian Action Network (DBAN) models from a small number of 2-D videos. This method computes the likelihood of a sampled pose by matching the foreground feature vectors computed over the projected human model with that obtained from the observed image. Simple pose matching metrics, such as foreground overlap [17, 20], have been popular in human action recognition literature due to their efficiency but are sensitive to foreground noise. Further, note that the matching is not straightforward, since the person scale and shape variations across different actors must also be taken into account. While local descriptors such as Shape Context [10] can be used for robust matching across shape variations, they are sensitive to small variations in blob shape and computing these descriptors is also computationally expensive. Another commonly used feature is optical flow [2, 6, 12] but obtained flows can be extremely noisy. Attempts have also been made to fit the sampled human pose to observations using stochastic search methods such as gradient descent and MCMC [8]. However, due to the non-convex nature of the likelihood function, gradient-based methods often fail due to local maxima and MCMC-based methods become computationally too expensive for efficient pose tracking. Recently, part-based graphical models [3] have been shown to accurately location 2D poses, but they do not model inter-part occlusion.

3 Action representation

Our action representation is based on the concept that a *composite* action can be decomposed into a sequence of simple *primitive* actions. Each primitive action *pe* modifies the *state* s of the actor to give a new state s'. For example, we consider *walking* as a composite action that involves four primitives—left leg forward → right leg crosses left leg → right leg forward → left leg crosses right leg. Each primitive can be defined as a conjunction of rotation of body parts, for e.g. during walking, rotation of upper leg about the hip and rotation of lower leg about the knee. To allow robust inference, we map these action models to a Dynamic Bayesian Network (DBAN).

3.1 Learning action models

We can learn the action models either from 2D pose and action boundary annotations [13, 16, 18] or from 3D Motion Capture sequence of the action [10], if available. To obtain the model for each composite action, [16] propose to manually select the keyposes for each action; each keypose marks a discontinuity in the angular representation of the human pose. The 3D model for each keypose is then obtained by lifting 3D pose from 2D annotations [13, 18]; alternatively if the MoCAP is available one may obtain the keyposes by computing pose energy [10]. These approaches either require expert knowledge to manually determine the keyposes or rely on the availability of MoCAP data for all the actions.

We propose to automatically determine the keyposes without the MoCAP data, by reconstructing approximate 3D

[1]Compared to [16], this paper includes additional explanations of the proposed framework with illustrative figures and an extended review of the related work.

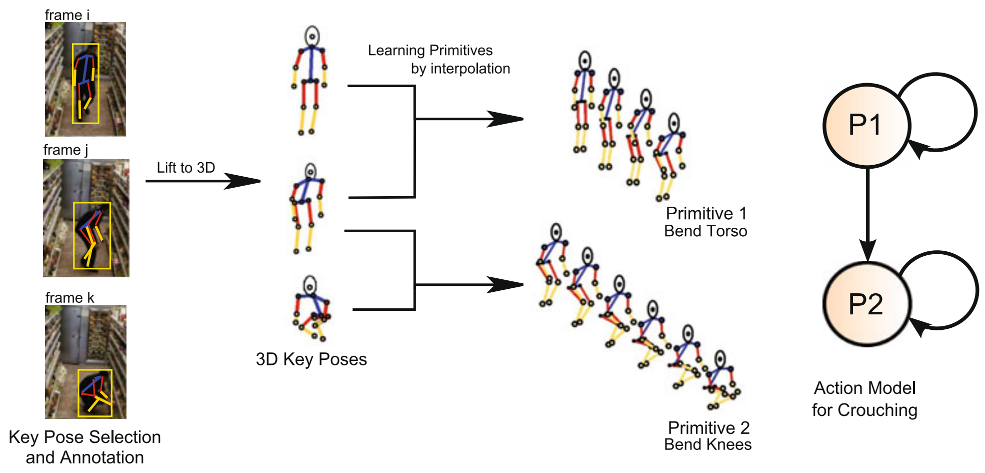

frame i
frame j
frame k
Key Pose Selection and Annotation
Lift to 3D
3D Key Poses
Learning Primitives by interpolation
Primitive 1 Bend Torso
Primitive 2 Bend Knees
P1
P2
Action Model for Crouching

Fig. 1 Action Model Illustration for *Crouch* action with three keyposes and two primitives (figure obtained from [13])

pose sequence from a few 2D annotations and then finding keyposes as peaks in the pose energy profile [10]. Given a video sequence of a composite action, we first uniformly annotate a few poses by marking the joints and the relative depths. These poses are then lifted to 3D using the approach described in [13, 16, 18]. We then transform the 3D joint position representation of the pose to the joint-angle representation, where the pose is represented by a length of each part and the relative angle of each part in the coordinate frame of the parent part. This transformation allows us to model the feasible range of the human pose using linear constraints. Let K denote the set of annotated poses, A denote the additional joint annotations (for smoothness). We formulate the 3D pose sequence recovery problem as the estimation of the joints angles $\boldsymbol{\theta}$ over all frames, given by (1):

$$\begin{aligned}&\sum_{f\in K}\sum_{j\in J}(\theta_{f,j}-\theta^a_{f,j})+\sum_{j\in A}\left\|\text{proj}(\theta_{f,j})-p^a_j\right\|^2\\&\quad+\sum_f(\boldsymbol{\theta}_{f+1}-\boldsymbol{\theta}_f)^2 \qquad (1)\\&\text{s.t.}\quad \forall f\quad \boldsymbol{\theta}^L<\boldsymbol{\theta}_f<\boldsymbol{\theta}^U\end{aligned}$$

where $\theta_{f,j}$ is the relative angle of joint j in frame k. The first term is the mismatch error between the estimation pose sequence and the annotated poses; the second term is the mismatch error in matching the additional annotations; and the final term guarantees that the pose change is slow and the estimated solution is unique. The vectors $\boldsymbol{\theta}^L$ and $\boldsymbol{\theta}^U$ are the lower and upper bounds on the human pose, to guarantee a feasible pose in each frame. We use levmar library [9] to solve (1).

At this stage, each composite action is essentially a sequence of 3D keyposes with time intervals. Now for every consecutive keypose pair, we define the primitive as the per time step transformation required to go from one keypose to the next i.e. the primitive transforms one keypose to next over a time duration. This duration model allows us to model the speed variations across multiple actors. Now since during a primitive, each part has rotated about a single axis, each primitive can be simply defined as a conjunction of the rotation of body parts. Note that, using this representation, we can obtain a strong prior on the 3D pose of a person performing a composite action, after time t has elapsed from the start of the action.

Figure 1 shows an illustration of action model obtained for the crouching action.

3.2 Dynamic Bayesian action network

Given the action models, [13] embeds them into a *Dynamic Bayesian Network (DBN)* which is referred to as the *Dynamic Bayesian Action Network (DBAN)*. DBAN used in [13] correspond to the first three layers on the model shown in Fig. 2, with foreground observation nodes (not drawn in the figure for clarity). The nodes in the topmost layer correspond to the composite actions like *walk*, *flap*, etc. The second layer corresponds to the primitives and the third layer corresponds to the human pose. A duration node is associated with each primitive that captures the time elapsed (in number of frames) since the primitive started. Thus, the *state* s_t of the DBAN at time t is denoted by the tuple (ce_t, pe_t, d_t, p_t). In this work, instead of directly evaluating the projection of 3D pose on the observations as in [13], we represent the projected pose using a 2D part model, which is represented as the fourth layer in Fig. 2.

The optimal state sequence $s^*_{[1:T]}$ for an observation sequence of length T is computed by maximizing the weighted sum of potentials similar to [1, 13]

$$s^*_{[1:T]}=\underset{\forall s_{[1:T]}}{\text{argmax}}\sum_{t=1}^{T}\left(\sum_f w_f\phi_f(s_t,I_t)+\psi(s_{t-1},s_t)\right) \qquad (2)$$

where $\phi_i(s_{t-1},s_t,I_t)$ are observation and transition potentials and w_i is the weight vector that models the relative importance of the potential functions. Note that DBAN is a

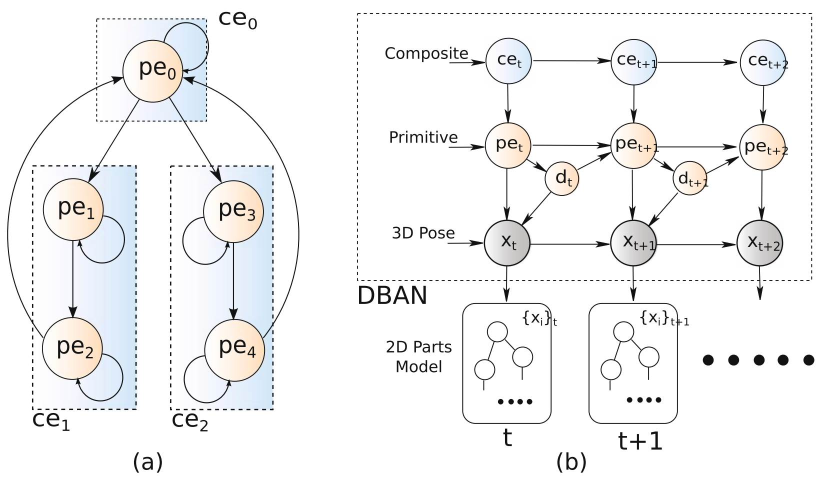


Fig. 2 Dynamic Bayesian Action Network with 2D Part Model (DBAN-Parts). (**a**) Rolled version showing Action network with two actions sharing primitives. (**b**) Unrolled version, DBAN is enclosed within the *dotted box*. Observation nodes are not shown for clarity

multi-variable state representation of the HMM in [1]. Further note that the objective function given by (2) has the same form in both DBAN-FGM and DBAN-Parts, however, the observation potentials are different since DBAN-Parts uses part models.

3.2.1 *Transition potential*

The state transitions are modeled by primitive and pose transition potentials, given by (3). The primitive transition potential captures speed variations across action instances by using *log* of *signum* function over the duration length, such that the probability of staying the same primitive pe_t decreases near the mean duration $\mu(pe_t)$ and the probability of transition to a new primitive increases.

The pose transition potential enforces smoothness over poses and is modeled using normal distribution with the mean and variance $\mathcal{N}(\theta_{\text{mean}}, \theta_{\text{var}})$ of displacement of body joints learned during the training

$$\psi(s_{t-1}, s_t) = \left(-\frac{((p_t - p_{t-1}) - \theta_{\text{mean}})^2}{2\theta_{\text{var}}^2}\right) + \left(-\ln\left(1 + e^{(-1+\beta\frac{d_t - \mu(pe_t)}{\sigma(pe_t)})}\right)\right) \tag{3}$$

where $\beta = 1$ if $pe_t = pe_{t-1}$, otherwise $\beta = -1$.

3.2.2 *Observation potential*

The observation potentials of a state $\phi_{\text{obs}}(s_t, o_t)$ are defined using features we extract from the video. DBAN-FGM [13] projects the 3D pose and computes the likelihood of the projected pose using multiple features, such as foreground overlap and difference image match. In this work, however, we project 3D pose to obtain a 2D part model and which allows an efficient local search and more accurate fit to the observation.

3.2.3 *Relative weight vector*

Reference [13] proposed a variant of *Voted Perceptron algorithm* [1] to learn the feature weights in a DBAN. The learning algorithm initializes with a random weight vector and iteratively update the weight vector to reduce the log likelihood error on the training data. In this work, we employ the same algorithm to the learn the weights.

4 Simultaneous pose tracking and action recognition

Here, we describe the algorithm to simultaneously track human pose and recognize the action in a video using DBAN-Parts. DBAN-FGM [13] infers the action label by matching all action models with observation sequence and finding the best match. Matching is done by sampling poses from action models and fitting the model to the observed image. For efficiency, instead of matching each action model separately and then selecting the best match, all models are matching simultaneously in one-pass by maintaining multiple state sequences. Since the number of possible sequences is combinatorial, all possible sequences cannot be considered. DBAN-FGM [13] uses a greedy strategy and maintains top N state sequences that have the highest score. This greedy selection step is too aggressive. If the number of samples N is small it often results in impoverishment of state samples from all actions thereby leaving samples only from one action after just a few frames; once all the samples from an action are pruned out, that action class is never reconsidered. If, however, the number of samples is too high, it drastically slows down the inference; furthermore, if after a few frames all state samples belong to the same action class, maintaining large number of samples as no benefit on accuracy and only hurts due to high computational cost.

In this work, we estimate the appropriate number of samples that needs to be maintained at each frame such that state samples from all likely actions are well represented; this is done by defining a measure of uncertainty over the currently active action labels. Below, we present step-by-step description of the proposed inference algorithm; the pseudo-code is shown in Algorithm 1.

Algorithm 1 Inference Algorithm

```
▷ Obtain initial states by sampling poses
  from all composite action models
  $S_1 = \{\langle s_0^{(i)}, \alpha_0^{(i)} \rangle \mid i = 1 \dots N_{\max}\}$
for $t = 0$ to $T$ do
    ▷ Obtain observation feature maps $O_{t+1}$
    for all $s_t^{(i)}$ do
        ▷ $d_{t+1} = d_t + 1$
        ▷ Obtain $\langle ce_{t+1}^{(i)}, pe_{t+1}^{(i)} \rangle \leftarrow \text{allow}(ce_t, pe_t, d_{t+1})$
        for all $\langle ce_{t+1}^{(i)}, pe_{t+1}^{(i)} \rangle$ do
            ▷ Sample pose from the action model,
              $p_{t+1}^{(i)} \sim \phi_p(p_t, pe_{t+1}, p_{t+1})$
            ▷ Compute the state potential
              $\alpha_{t+1}^{(i)} = \alpha_t^{(i)} + \sum_f w_f \phi_f(s_t^{(i)}, s_{t+1}, o_{t+1})$
            ▷ Push $\langle s_{t+1}, \alpha \rangle$ to $S_{t+1}$
        end for
    end for
    ▷ Obtain action class likelihood vector,
      $v = \{v_{ce}\}$, where $v_{ce} = \max_{s_{t+1}^{(i)} = \langle ce, \dots \rangle} \alpha_{t+1}^{(i)}$
    ▷ Set target sample set size, $N_t \propto \left(\sum_{ce} v_{ce} \log(v_{ce})\right) \times N_{\max}$
    ▷ Prune $S_{t+1}$ such that $|S_{t+1}| \leftarrow \max(N_t, N_{\min})$
end for
▷ actionlabel = $\arg\max_{s_T^{(i)}} \alpha_T^{(i)}$
```

4.1 Initialization

For initializing the state distribution, we sample poses from all the composite actions in the action set. For viewpoint invariance, all likely viewpoints are considered for each pose sample from every composite action model.

4.2 Prediction: sample next state

For each state s_t, we increment the duration of the state by unit time step. Given the current action (ce_t, pe_t) and new duration, we then sample the next action state (ce_{t+1}, pe_{t+1}). Note that if primitive transition occurs, then the duration is set to 0 to mark the start of a new primitive. Next, we sample from the *pose transition potential* $\phi_p(p_t, pe_{t+1}, p_{t+1})$ to choose the next pose p_{t+1}.

4.3 Fit the sampled state to the observation

We first apply a pedestrian detector [21] to find the person in the video, thus our algorithm initializes only when a standing pose is observed. We then apply a combined shape and foreground blob tracker to track and localize the person in each frame, even through changing poses. The position and scale information available from the person tracker is then used to adjust the 3D pose sampled from the action model in the previous step. Given the adjusted 3D pose, we then orthographically project the pose to construct a 2D part model which is then used for accurate localization. Note that during the projection step, we automatically determine the non-observable/occluded parts and do not use those parts for localization. Figure 3 show some sample 3D poses and corresponding 2D part models. Using the 2D part model, we then perform a local search to accurately fit the pose to the observation. The details on obtaining the 2D part model from 3D pose and the local search is described in detail in Sect. 4. The likelihood of the pose/state is then computed by matching the localized 2D pose with low-level image features. This includes computing the observation potential $\phi_{obs}(s_{t+1}, o_{t+1})$ using foreground match, difference image and part templates (described later in Sect. 4.2).

4.4 Selecting the state samples

Since maintaining all possible state sequences is not possible, only a small number of states are retained in each frame. As discussed earlier, a greedy sample selection step can lead to sample impoverishment and may significantly affect both accuracy and efficiency of the algorithm. To avoid action sample impoverishment, we set the minimum number samples $N_{\min}$ to be maintained in each frame; we also set $N_{\max}^a$ as the maximum number of samples allowed for any action class. Note that this may address the sample impoverishment from different action classes but still has poor efficiency.

We define a measure of action label uncertainty in the current frame by computing the entropy over the distribution of currently active actions/states; an action is considered *active*, if there is a state sample corresponding to that action. To compute the entropy of the currently active actions, we compute the action class likelihood vector $v = \{v_{ce}\}$, where v_{ce} is the highest likelihood score over all states in the current frame that belong to the action class ce. Given the action class likelihood vector v, we then define the target sample set size N_t in the current frame t as

$$N_t \propto \left(\sum_{ce} v_{ce} \log(v_{ce}) \right) \times N_{\max}. \tag{4}$$

Note that when action label uncertainty is high (i.e. state samples corresponding to different action classes have similar scores), large number of samples (N_t is high) are maintained thereby allowing presence of samples from different action classes and avoiding impoverishment. When uncertainty is low i.e. the samples are likely to the belong to the same/few action class, and thus only a few samples are enough for accurate inference; note that maintaining fewer samples also speeds up the inference.

5 3D Pose observation using 2D body part model

In this section, we describe the localization of a 3D pose projected from a given viewpoint. This is achieved using a graphical model of the 2D body parts. The body part model used in the work is similar to the *Pictorial Structures* [3] which is widely used for estimating human pose in an image. The model has 10 nodes, each corresponding to a body part—head, torso, upper arms (l, r), lower arms (l, r), upper legs (l, r) and lower legs (l, r). These nodes are connected with edges that capture the kinematic relationship between the parts. Figures 3 and 4 show the part model. For localization using *Pictorial structure*, the individual parts are searched by applying part detectors at all locations and orientations followed by belief propagation to enforce kinematic constraints. Note, however, that in this work, our objective is to accurately localize a 3D pose projected from a given viewpoint. This imposes a strong constraint on the orientation of the body parts and their kinematic relationship. Furthermore, approximate position and scale information is also available from the person detection step. Hence, localization in our case does not require a dense part search. However, during localization, we need to tackle the problem that some of the body parts may not be observable, either due to inter-part occlusion (see Fig. 4) or 3D–2D projection (see Fig. 3(b)).

Now we first describe the process to generate the 2D part model appropriate for localization, and then we briefly describe the localization using local search.

5.1 Building 2D part models

We project the 3D pose from the given viewpoint to estimate the 2D position of the body joints. From the body joints, we build a rectangular cardboard model by fitting a rectangle between every pair of joints connected by a body part (shown in Fig. 4(c)). During projection, we also estimate the relative depth order of the 2D parts. Next, we determine which parts are visible based on the depth order and pairwise overlap between the part rectangles. In our experiments, we considered parts with percentage visibility below 50% to be occluded. Furthermore, when projecting 3D pose to 2D, some of the body parts are too small to be observed (see Fig. 3(b)) and thus are not useful for localization. Figure 4 shows the flowchart of building the 2D part model for a 3D pose from a given viewpoint.

5.2 Part detectors

We use template-based matching for detecting body parts in the image. We model the head with an ellipse template, torso with an oriented rectangle and each arm with a foreshortened pair of lines. We compute the log likelihood score $\phi(I|x)$ of a part hypothesis x by accumulating the edge strength and orientation match scores on the boundary points.

$$\phi(I|x) = \sum_{x_i \in x} d_{\text{mag}}\big(I(x_i)\big) \times d_{\text{ori}}\big(x_i, I(x_i)\big) \tag{5}$$

where $d_{\text{mag}}(I(p))$ is the approximate Euclidean distance to the nearest edge pixel from the image point p, weighted by the edge strength. This can be calculated very efficiently using generalized distance transform over the edge likelihood map [3]; $d_{\text{ori}}(p, I(p))$ is the orientation likelihood, which is the dot product between the normals at the model point p and corresponding point in the image $I(p)$. Since the orientation information is often very noisy, we approximate the normals by quantizing into eight bin orientations.

5.3 Localization using 2D part models

Given the scale adjusted 3D pose X and position information, we first apply the detector for each part x_i over the expected position and orientation of the part and a small neighborhood around it. Pose estimate is then obtained by

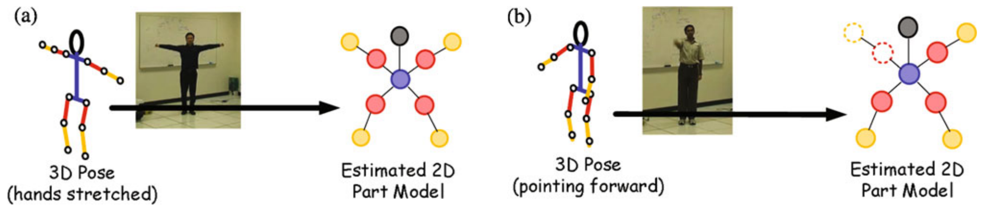


Fig. 3 Sample 3D poses and their corresponding 2D part models used for alignment; non-observable nodes in (**b**) are marked with dotted boundary

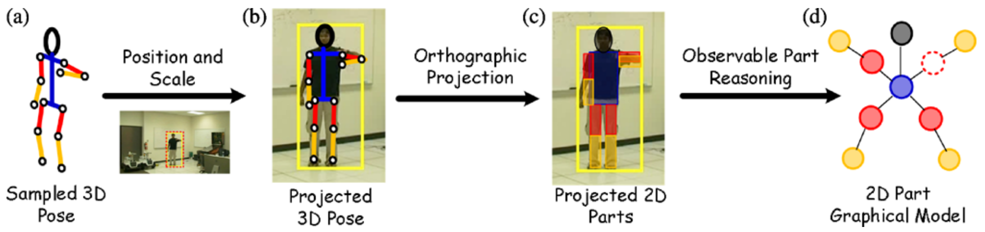


Fig. 4 Illustration of estimating 2D Body part models from the 3D Pose; Note that the node corresponding to right upper arm is non-observable (due to occlusion) and is marked with dotted boundary in (**d**)

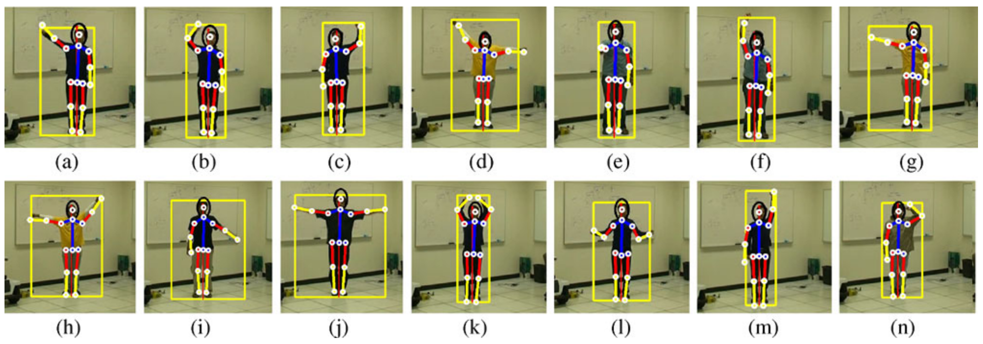


Fig. 5 Results on the Gesture dataset: the bounding box shows the person position and the estimated pose is overlaid on top of the image, illustrated by limb axes and joints

maximizing the average log likelihood of the visible parts, given by

$$\phi_{\text{ps}}(p, I) = \sum_{i \in V} \phi_i(I|x_i) + \sum_{ij \in E} \psi_{ij}(x_i, x_j) \tag{6}$$

where V is the set of all visible body parts, E is the set of part pairs that are kinematically connected; and $\phi_i()$ is the likelihood map of part i obtained by applying the detector. We normalize the total potential by number of visible parts to remove bias toward poses with fewer visible parts. Note that occlusion sensitive pose localization proposed here is different from that used proposed in [15]. Compared to [15] which consider pixel level occlusion constraints for each part, we only consider parts which are almost completely visible (with visibility more than 80%). This allows our inference to be much more efficient and as we show in our results, the localization is accurate enough for reliable action recognition.

6 Experiments

We tested our approach on the Hand Gesture Dataset [13]; the dataset includes about 500 instances of hand gestures used in HCI applications.

[Dataset Description]
The Gesture dataset has 12 actions performed by eight different actors, captured from a static camera in an indoor lab setting. The action set include—Column Left (bend left arm from side to overhead), Column Right, Open Up (move both arms from overhead to side), Close Up (move both arms upward from side), Turn Right (extend arm to right side), Turn Left, Line (extend arms parallel to ground), Close Distance (clap), Stop Right (raise right arm upward to the full extent of arm), Stop Left, Action Left (extend both arms, then raise left arm overhead), Action Right. Figure 5 shows sample frames from the dataset.

Each action sequence in the dataset has exactly one person performing the action, facing the camera. Each actor performs every action about five times, thus the dataset contains a total of about 500 action sequences. The videos are 852 × 480 resolution, and the height of person varies between 200–250 *pixels* across different actors. This set is similar to that used in [14] but has a bigger variety. As the background is not highly cluttered, extracted foreground is quite accurate but the large number of actions with subtle differences makes recognition still a challenging task.

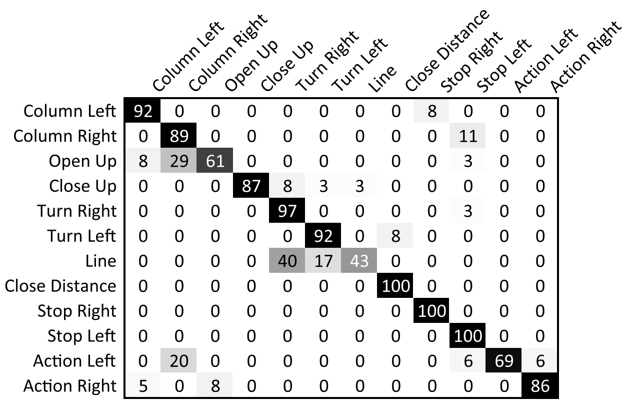
[Experiment Settings]
To compare our inference algorithm with that in [13], we use the same experiment settings wherever possible. The models for each action were obtained by video annotation. For learning the feature weights for each action model, the same training data as on which the action model is trained. The feature weights were randomly initialized and the one that achieves the highest accuracy on the training set was used during testing.

During inference, we set the minimum samples for each action N_{min} to three and maximum number of samples in any frame N_{max} to 15. In our experiments, the actual number of samples in a frame varied between three and 15 due to the entropy-based sample set selection, and on an average about seven samples were maintained in each frame.

[Quantitative Evaluation]
To evaluate the performance of our approach, we computed both the action classification accuracy and the error in pose estimates. We split the action sequences into train and test sets based on the actors i.e. the action models trained on a subset of actors and test on the rest. Since each video sequence contains only one action; it is said to be recognized correctly if its label is the same as in the ground truth. Since the primary contribution of DBAN-FGM [13] is on learning action models with low training requirements, we ran our experiments with train:test ratio of 1:7. The second column of Table 1 provides the recognition results, averaged two training sets. The accuracy numbers for 1:7 train:test for DBAN-FGM were obtained from Fig. 6(b) in [13].

Table 1 Performance on Gesture Store dataset

Approach	Train:Test ratio	Recognition (% accuracy)	2D Tracking (% accuracy)
DBAN-FGM [13]	1:7	78.6	75.67 (89.94)
DBAN-Parts	1 : 7	**84.52**	**81.76** (92.66)



	Column Left	Column Right	Open Up	Close Up	Turn Right	Turn Left	Line	Close Distance	Stop Right	Stop Left	Action Left	Action Right
Column Left	92	0	0	0	0	0	0	0	8	0	0	0
Column Right	0	89	0	0	0	0	0	0	0	11	0	0
Open Up	8	29	61	0	0	0	0	0	0	3	0	0
Close Up	0	0	0	87	8	3	3	0	0	0	0	0
Turn Right	0	0	0	0	97	0	0	0	0	3	0	0
Turn Left	0	0	0	0	0	92	0	8	0	0	0	0
Line	0	0	0	0	40	17	43	0	0	0	0	0
Close Distance	0	0	0	0	0	0	0	100	0	0	0	0
Stop Right	0	0	0	0	0	0	0	0	100	0	0	0
Stop Left	0	0	0	0	0	0	0	0	0	100	0	0
Action Left	0	20	0	0	0	0	0	0	0	6	69	6
Action Right	5	0	8	0	0	0	0	0	0	0	0	86

Fig. 6 Confusion matrix for the Hand Gesture dataset

To evaluate the effect of our 2D Part model, we evaluate the errors in the pose estimate obtained using our inference with DBAN-FGM (that does not use 2D part model). To measure the error in pose estimates, we manually annotated 48 2D poses four randomly selected frames from an instance of each action class, and compute the accuracy of the estimated 2D parts. A 2D part estimate is considered correct if it lies within the length of the ground truth segment. Since our experiments are on hand gestures, a more meaningful evaluation is to compute the pose accuracy only over the arms, since arms are the only parts involved in the action. The third column of Table 1 provides the pose accuracy computed over the arms (192 annotations); the numbers within parentheses show the accuracy over all the body parts (480 annotations). Even though the improvement in pose accuracy averaged over all parts is not quite significant (only 2.5%), notice that the accuracy over the parts relevant to the action (arms) is about 6%.

We also report the confusion matrix for the recognizing actions using DBAN-Parts over the entire dataset. Figure 6 shows the confusion matrix for train:test ratio of 1:7. Notice that the recognition accuracy is around 85–90% for each action, except for *Line* and *OpenUp* actions which got misclassified as *TurnRight* and *ColumnRight*, respectively. Observe that in both cases, action model of one arm is same and hence the confusion is expected due to similarity in poses. We believe more accurate part detectors would be able to localize better and deal with such ambiguities. Further note that the actions (*CloseDistance*, *StopRight* and *StopLeft*) that contain poses where arms are non-observable/occluded, get correctly recognized; Fig. 5(e) shows an example of such a pose in *StopRight* action, where the right arm is occluded.

7 Conclusion

In this work, we have presented a general framework for simultaneous tracking and action recognition using 2D part models with Dynamic Bayesian Action Network [13]. The 2D part model allows more accurate pose alignment with the observations, thereby improving the recognition accuracy. To compensate for the additional time required for 2D part alignment, we proposed an action entropy-based scheme to determine the minimum number of samples to be maintained in each frame while avoiding sample impoverishment.

In future, we plan to apply this algorithm in more complex domains with cluttered environments by employing more accurate part detectors. In existing implementations, action models are defined by dynamics over all the body parts; however, by defining action models only over the parts that are involved in the action, our framework can be extended to recognize larger number of actions [5]. Note that the proposed framework can be easily extended to deal with multiple actors by incorporating actor-id in the state definition.

References

1. Collins, M.: Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), July 2002, pp. 1–8 (2002)
2. Fathi, A., Mori, G.: Action recognition by learning mid-level motion features. In: Computer Vision and Pattern Recognition (CVPR) (2008)
3. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. Int. J. Comput. Vis. **61**(1), 55–79 (2005)
4. Gupta, A., Chen, F., Kimber, D., Davis, L.S.: Context and observation driven latent variable model for human pose estimation. In: Computer Vision and Pattern Recognition (CVPR) (2008)
5. Ikizler, N., Forsyth, D.A.: Searching video for complex activities with finite state models. In: Computer Vision and Pattern Recognition (CVPR) (2007)
6. Ke, Y., Sukthankar, R., Hebert, M.: Event detection in crowded videos. In: International Conference on Computer Vision (ICCV) (2007)
7. Laptev, I.: On space-time interest points. Int. J. Comput. Vis. **64**(2–3), 107–123 (2005)
8. Lee, M.W., Nevatia, R.: Human pose tracking using multi-level structured models. In: ECCV (3), pp. 368–381 (2006)
9. Lourakis, M.: Levmar: Levenberg-Marquardt nonlinear least squares algorithms in C/C++. [web page]. http://www.ics.forth.gr/~lourakis/levmar/, Jul. 2004. [Accessed on 31 Jan. 2005.]
10. Lv, F., Nevatia, R.: Single view human action recognition using key pose matching and Viterbi path searching. In: Computer Vision and Pattern Recognition (CVPR) (2007)

11. Morency, L.-P., Quattoni, A., Darrell, T.: Latent-dynamic discriminative models for continuous gesture recognition. In: Computer Vision and Pattern Recognition (2007)
12. Natarajan, P., Nevatia, R.: View and scale invariant action recognition using multiview shape-flow models. In: CVPR (2008)
13. Natarajan, P., Singh, V.K., Nevatia, R.: Learning 3d action models from a few 2d videos for view invariant action recognition. In: CVPR (2010)
14. Shet, V., Prasad, S.N., Elgammal, A., Yacoob, Y., Davis, L.: Multi-cue exemplar-based nonparametric model for gesture recognition. In: ICVGIP (2004)
15. Sigal, L., Black, M.J.: Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In: CVPR, pp. 2041–2048 (2006)
16. Singh, V.K., Nevatia, R.: Human action recognition using a dynamic Bayesian action network with 2D part models. In: Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP '10, pp. 17–24 (2010)
17. Sminchisescu, C., Kanaujia, A., Li, Z., Metaxas, D.: Conditional random fields for contextual human motion recognition. In: International Conference on Computer Vision (ICCV), pp. 1808–1815 (2005)
18. Taylor, C.J.: Reconstruction of articulated objects from point correspondences in a single uncalibrated image. In: Computer Vision and Image Understanding (CVIU), vol. 80, pp. 349–363 (2000)
19. Urtasun, R., Fleet, D.J., Fua, P.: 3D people tracking with Gaussian process dynamical models. In: Computer Vision and Pattern Recognition (CVPR), pp. 238–245 (2006)
20. Weinland, D., Ronfard, R., Boyer, E.: Automatic discovery of action taxonomies from multiple views. In: Computer Vision and Pattern Recognition (CVPR), vol. II, pp. 1639–1645 (2006)
21. Wu, B., Nevatia, R.: Detection of multiple, partially occluded humans in a single image by Bayesian combination of Edgelet part detectors. In: ICCV, pp. 90–97 (2005)

Vivek Kumar Singh received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Kanpur, in 2005, and the M.S. degree in computer science in 2007 from the University of Southern California (USC), where he is currently pursuing the Ph.D. degree in the Department of Computer Science. His research interests include computer vision, machine learning, optimization, and his current research focuses on human pose estimation and action recognition.

Ram Nevatia received the B.S. degree in electrical engineering from the University of Bombay, India, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Palo Alto, California. He has been with the University of Southern California, Los Angeles, since 1975, where he is currently a professor of computer science and electrical engineering and the director of the Institute for Robotics and Intelligent Systems. His research interests include computer vision, artificial intelligence, and robotics. He is the author of two books and has contributed chapters to several others. He is a fellow of the AAAI and the IEEE.