Randomized Trees for Human Pose Detection

Grégory Rogez[1], Jonathan Rihan[2], Srikumar Ramalingam[2], Carlos Orrite[1] and Philip H.S. Torr[2]

[1]Computer Vision Lab - I3A
University of Zaragoza, SPAIN
{grogez, corrite}@unizar.es
http://www.cv.i3a.unizar.es

[2]Department of Computing
Oxford Brookes University, UK
{jon.rihan,srikumar.ramalingam,philiptorr}@brookes.ac.uk
http://cms.brookes.ac.uk/research/visiongroup/

Abstract

This paper addresses human pose recognition from video sequences by formulating it as a classification problem. Unlike much previous work we do not make any assumptions on the availability of clean segmentation. The first step of this work consists in a novel method of aligning the training images using 3D Mocap data. Next we define classes by discretizing a 2D manifold whose two dimensions are camera viewpoint and actions. Our main contribution is a pose detection algorithm based on random forests. A bottom-up approach is followed to build a decision tree by recursively clustering and merging the classes at each level. For each node of the decision tree we build a list of potentially discriminative features using the alignment of training images; in this paper we consider Histograms of Orientated Gradient (HOG). We finally grow an ensemble of trees by randomly sampling one of the selected HOG blocks at each node. Our proposed approach gives promising results with both fixed and moving cameras.

1. Introduction

Full-body human pose recognition from monocular images constitutes one of the fundamental problems in Computer Vision. It has a wide range of potential applications such as human/computer interfaces, video-games or surveillance. Given an input image, an ideal system would be able, first, to localize any humans present in the scene and second to recover their poses. The two stages, known as *human detection* and *human pose recognition*, are usually considered separately. There is an extensive literature on both *detection* [9, 12, 27, 25, 26, 18] and *recognition* [1, 16, 19, 4, 17, 22] but relatively few papers consider the two stages together [10, 3]. Most algorithms for pose recognition assume that the human has been localized and the silhouette has been recovered, making the problem substantially easier.

In this work, we propose an efficient method to jointly localize and recognize the pose of humans, using an exemplar based approach and fast search technique. Such pose detector would be very useful for initializing model-based-approaches[17], tracking algorithms [24] or segmentation algorithms [7].

1.1. Related Previous Work

Exemplar based approaches have been very successful in pose recognition [16]. However, in a scenario involving a wide range of viewpoints and poses, a large number of exemplars would be required. As a result the computational time would be very high to recognize individual poses. One approach, based on efficient nearest neighbours search using histogram of gradient features, addressed the problem of quick retrieval in large set of exemplars by using Parameters Sensitive Hashing (PSH) [19], a variant of the original Locality Sensitive Hashing algorithm. The final pose estimate is then produced by locally-weighted regression which uses the neighbours found by PSH to dynamically build a model of the neighbourhood.

The method of Agarwal and Triggs [1] is also exemplar based, they also use a kernel based regression but they do not perform a nearest neighbors search for exemplars, instead using a hopefully sparse subset of the exemplars learnt by the RVM. Their method has the main disadvantage that it is silhouette based, perhaps, more serious it can not model ambiguity in pose as the regression is unimodal. In [12], Gavrila presents a probabilistic approach to hierarchical, exemplar-based shape matching. This method achieves a very good detection rate and real time performance but does not regress to a pose estimation.

In [10], the authors present a template-based pose detector and solve the problem of huge dataset by detecting only human silhouette in a characteristic postures (sideways opened-leg walking postures in this case). They recently extended this work in [11] by inferring 3D poses between consecutive detections using motion models. This work

978-1-4244-2243-2/08/$25.00

gave some very interesting results with moving cameras. However it seems somehow difficult to generalize to any actions that do not exhibit characteristic posture. In [23], an exemplar-based approach with dynamics is proposed for tracking pedestrians.

In this work, we propose to solve human pose recognition problem by formulating it as a classification task. Our pose detection algorithm uses the best components of hierarchical trees as well as randomized forests. Indeed, Randomized trees [2] and Random Forests [8] have shown to be fast and robust classification techniques that can handle multi-class problems [14]. More recently Bosh et al. used Random forests for object recognition [6].

Many different types of features have been considered for human detection and pose recognition: silhouette [1], shape [12], edges [10], HOG descriptors [9, 27, 19], Haar filters [25], motion and appearance patches [4], edgelet feature [26], shapelet features [18] or SIFT [15]. Driven by the recent success of HOG descriptors for both human detection [9, 27] and pose recognition [19], we chose to use them in our work.

1.2. Overview of the Approach

We consider the problem of detecting people and recognizing their poses at the same time as in [3] or [21] for hands, and propose using the HumanEva data set [20] for both training and testing as in [4]. In this work we focus on walking motion sequences, although our algorithm can be generalized for any action. Some work on pose recognition assumes that the bounding box is provided (e.g. [4]), we will consider a multi-stage algorithm and try to combine detection, classification and pose estimation in a cascade algorithm as shown in the diagram presented in Figure 1.

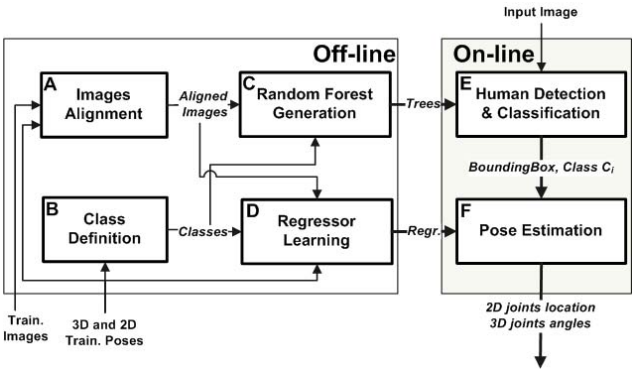


Figure 1. ***Systems Diagram.*** *The off-line stage can be divided into 2 types of processes: the pre-processing blocks (A and B) and the learning Blocks (C and D). The on-line algorithm comprises the detection and classification block E and the regression block F.*

Our first contribution is a novel method to improve the feature relevance by aligning the training images using 3D Mocap data (block A). Next we define the classes by discretizing a 2D manifold whose two dimensions are camera viewpoint and action (block B). We combine ideas from hierarchical clustering [12] and from [10] to select potentially discriminative feature when building a hierarchical decision tree. We thus propose growing a Random Forest by randomly sampling one of the selected discriminative feature at each node of the decision tree (block C). We then learn a series of regressors (block D) to estimate the 3D pose. The on-line stage is then made of the pose detection (block E) and pose estimation (block F).

The structure of the rest of the paper is as follows. In § 2, we introduce the pre-processing steps. In § 3, we detail the generation of our Random Forest. In § 4, we detail our pose detection algorithm. Some results are then presented in § 5 and some conclusions are finally drawn in § 6.

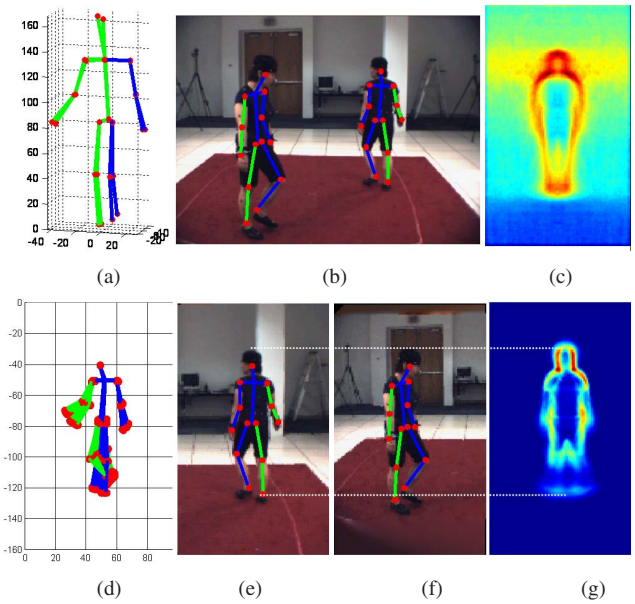


Figure 2. ***Alignment of training images.*** *(a) Training 3D poses from Mocap data. (b) Training images with projected 2D Poses. (c) Average gradient image over INRIA training examples (96x160). (d) Aligned and re-sized 2D poses (96x160). (e) and (f) cropped images (96x160) with normalized 2D poses. (g) Average gradient image over aligned HumanEva [20] training examples.*

2. Pre-processing Steps

As mentioned before, the first two contributions of this work appear in pre-processing the training data. Firstly, we aim to establish strong correspondences between all the training images by aligning them automatically, thus making the selection of useful features much easier. Secondly, we need to define classes to train our pose classifier.

2.1. Alignment of training data

While other approaches require a manual process [9] or a clean silhouette shape (either synthetic [19, 1, 10] or from

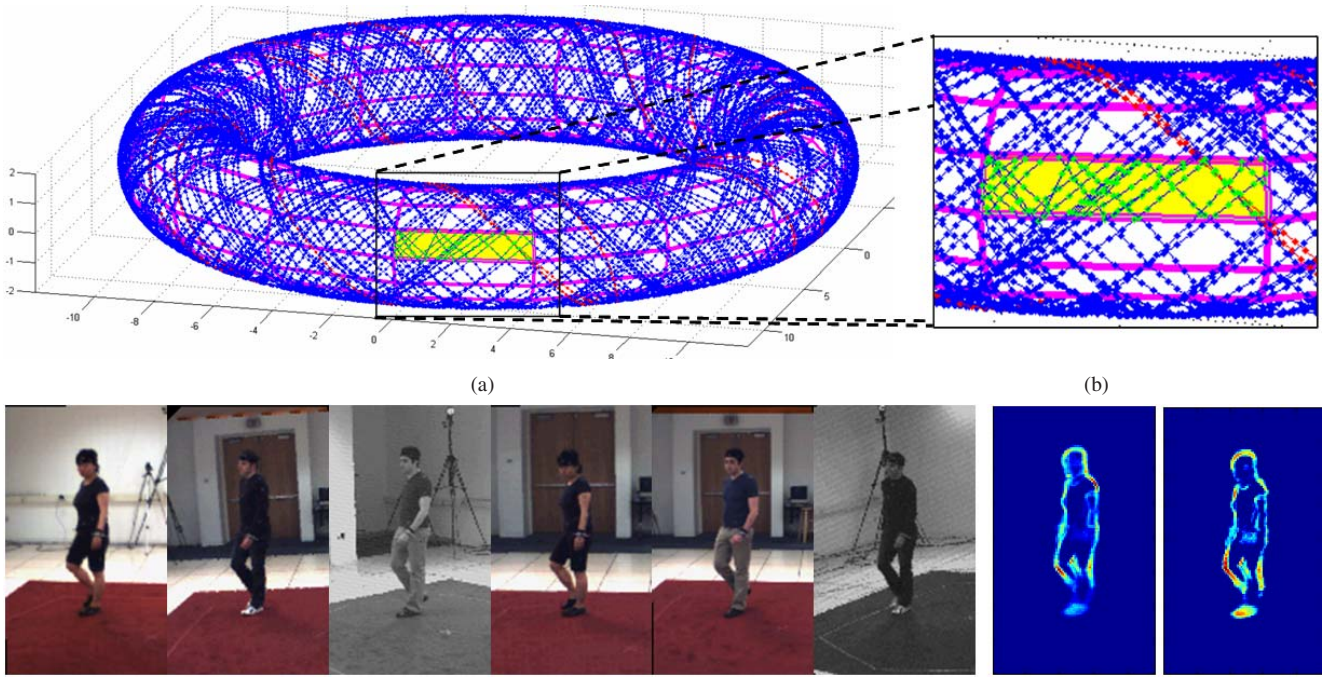


Figure 3. ***Class definition.*** *Top row (a): Torus manifold (see § 2.2) with discrete set of classes and training sequences (each blue dot represents a training image). (b) Zoom on a particular class (yellow) with corresponding training images (green dots). Bottom row from left to right: 6 Examples of aligned images belonging to the class highlighted in (b) for different subjects and different camera views, resulting average gradient and Log-likelihood ratio for this same class vs all the other classes. We provide as supplemental material a video showing the Log-likelihood for all the classes.*

manual labelling [12, 17]) for feature alignment, we propose a novel fully automatic process for accurately aligning real training images. We make the assumption that the 3D Mocap data (3D poses) corresponding to the training images are available. We then exploit a simple but effective way of taking advantage of their 2D projections in the image plane for aligning the images.

In more detail, the 3D joints of every training pose are first projected onto the corresponding image plane. The resulting 2D joints are then aligned using rigid body Procrustes alignment [5], uniformly scaled and centered in a reference bounding-box. For each training pose, we then estimate the four parameters corresponding to a similarity transformation (one for rotation, two degrees of freedom for translation and one scale parameter) between the original 2D joints locations in the original input image and the corresponding aligned and re-sized 2D joints. This transformation is finally applied to the original image leading to the cropped and aligned image. In this work, we chose to normalize all the training images to 96x160 as in [9].

The data set (images and poses) is flipped along the horizontal direction to double the training data size. By this process, and considering the HumanEva I data set (3 Subjects and 7 camera views) for training, we generate a huge data set[1] of more than 40,000 aligned and normalized 96x160 images of walking people, with corresponding 2D and 3D Poses. The complete process is depicted in Fig. 2. Note that the average gradient image obtained with our training data set shows more variability in the lower region compared to the one obtained from INRIA data set. This is due to the fact that most of the INRIA images present standing people.

[1]This data set will be publicly available soon.

2.2. Class definition

We intend to discretize the pose space into a distinct set of classes. Class definition is not a trivial problem because changes in both the human motion as well as the viewpoint are continuous and not discrete. In other words, it is not trivial to decide where a class ends and where the next one starts. In [19], the authors define the pose neighbors based on the distance between 3D joints. This definition produced good results in the absence of viewpoint changes. However, two poses which are exactly the same in the pose space could still have completely different appearances in the images due to changes in viewpoint. Thus it is critical to consider viewpoint information in the class definition.

We propose the use of a 2D manifold representation where the action (consecutive 3D poses) is represented by a 1D manifold and the viewpoint by another 1D manifold. Because of the cyclicity of the viewpoint parameter, if we model it with a circle the resulting manifold is in fact a cylindrical one. When the action is cyclic too, as with gait, jog etc., the resulting 2D manifold lies on a “closed cylin-

der" topologically equivalent to a torus [13, 17]. The classes definition task then takes the following steps:

1. "Align the gait sequences temporally" as proposed in [24] and map them to a torus manifold (cf [13])
2. Define the classes on this torus manifold by discretizing both the gait cycle and the viewpoint (Ng x Nv = N classes). See Fig. 3. In this paper, we chose to define 12x16=192 classes and let for future work the estimation of the optimum number of classes.

This definition of the classes on a torus leads to the creation of the toroidal Probability Transition Matrix [17] that can solve ambiguities and misclassification issues: along a sequence the most probable classes can be selected using spatio-temporal constraints. Because of the continuity of the motion and viewpoint, adjacent cells should not be considered as misclassification to solve some potential issues on the "boundaries of each class". We thus propose defining the misclassification based on an 8-neighbors similarity on the toroidal matrix.

3. Random Forest Generation

As described in [8], Random Forests work as follows. Given a set of training examples, a set of random trees H is created such that for the k-th tree in the forest, a random vector Φ_k is generated independently of the past random vectors $\Phi_1..\Phi_{k-1}$. This vector Φ_k is then used to grow the tree resulting in a classifier $h_k(\mathbf{x}, \Phi_k)$ where $\mathbf{x}$ is a feature vector. For each tree, a decision function splits the training data that reach a node at a given level in the tree.

The resulting forest classifier H is used to classify a given feature vector $\mathbf{x}$ by taking the mode of all the classifications made by the tree classifiers $h \in H$ in the forest.

3.1. Selection of Discriminative features

It is very important to select the relevant and most informative features in order to alleviate the effects of the curse of dimensionality. Many different features are used in general recognition problems. However, only few of them are useful in the given exemplar-based pose recognition task. For example, features like color and texture are very informative in general recognition problems, because of their variation due to clothing and lighting conditions, they are seldom useful in exemplar-based pose recognition. On the other hand, gradients and edges are more robust cues with respect to clothing and lighting variations [2]. Thus, after much experimentation, we chose to build our feature vector using HOG descriptors [9, 27, 19].

[2]Note that clothing could still be a problem for edges if there are very few subjects in the training set: some edges due to clothing (and not due to the pose) could be considered as discriminative edges when they should not be.

The usage of HOG blocks over the entire image leads to a very large feature vector. Thus an important question is how to select the most informative HOG blocks to use in the feature vector. Some works have addressed this question for human detection and pose recognition problems using SVM's or RVM's [9, 27, 3]. However, such learning methods are computationally inefficient for huge data sets.

In [10], the authors use statistical learning techniques during the training phase to estimate and store the relevance of the different silhouette parts to the recognition task. We use a similar idea to learn relevant features, although slightly different because of the absence of silhouette information. In what follows, we present our method to select the most informative HOG blocks. The basic idea is to take advantage of the accurate image alignment, that we achieved in section 2.1, and locate the most informative HOG blocks. In order to do this we study edge distribution over the entire training set $p(E)$, and favor locations that we expect to be more discriminative between different classes (similar in spirit to [19]). We thus construct intra and inter-class probability density maps of edge distribution that can be used to select the right features: the log-likelihood ratios give information on how discriminative the features are, based on their location. Here we describe a simple Bayesian formulation to compute the log-likelihood ratios which will give us the importance of different regions in the image. Let C_1, C_2,..C_n be n different classes. Let the probability for the class C_i, given the complete edge map, be $p(C_i|E)$. Using simple Bayes rule we have the following:

$$p(C_i|E) = \frac{p(E|C_i)p(C_i)}{p(E)} \tag{1}$$

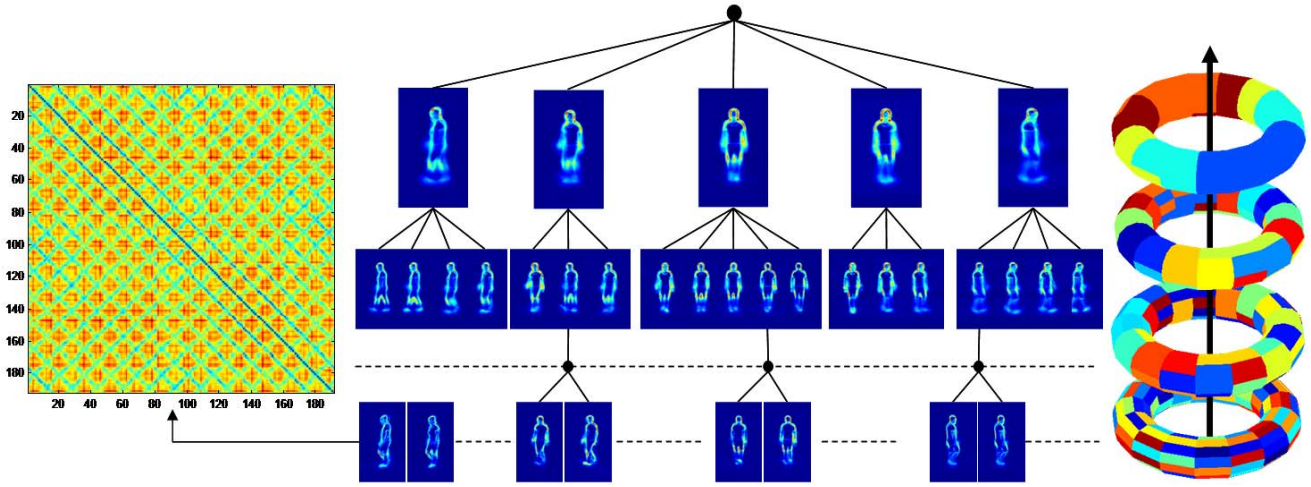
We present the log-likelihood ratio for the i_{th}class:

$$L_i = \log(\frac{p(E, C_i)}{p(E)}) \tag{2}$$

We compute $p(E, C_i)$ using the edge maps that correspond only to the training images belonging to class C_i. $p(E)$ is computed using all the training images corresponding to classes C_1 to C_n. Given this log-likelihood ratio, we can randomly sample boxes from positions where they are expected to be useful and reduce the dimension of the feature space considering only the discriminative HOG blocks. For the example given in Fig. 3, we can observe how the right knee is a very discriminative region for this class.

3.2. Bottom-up Hierarchical Tree learning

A hierarchical tree is then built using a bottom-up approach by recursively clustering and merging the classes based on a similarity matrix that is recomputed at each depth of the tree (Fig. 4). By "hierarchical" , we mean that the similarity between compared classes increases and that



(a) (b) (c)

Figure 4. ***Bottom-up Hierarchical Tree learning.*** *The hierarchical tree is built using a bottom-up approach by recursively clustering and merging the classes at each level. The similarity matrix is computed using the L2 distance between the log-likelihood ratios of the classes. The matrix presented here (a) is built from the initial 192 classes and used to merge the classes at the very lowest level of the tree. The similarity matrix is then recomputed at each level of the tree with the resulting new classes. The resulting tree is shown in (b) while the merging process on the torus manifold is depicted in (c).*

the classification gets more difficult when going down the tree and reaching lower levels.

At each depth of the tree, we repeat the following steps:

1. Compute new edge maps with the subclasses belonging to each cluster

2. Compute log-Likelihood ratios L_i for the new clusters/classes

3. Compute similarity matrix over the clusters at this depth using L2 distance between L_i

4. Merge classes based on similarity and define new classes

This process leads to the hierarchical structure represented in Fig. 4.

3.3. Random Selection of Discriminative Features

HOG descriptors need only to be placed near areas of high edge probability for a particular class. We then select the most discriminative (HOG) features at each node using the log-likelihood ratio as explained in section 3.1. This log-likelihood ratio is recomputed for each child at each node of the tree. Features are then extracted from all positive and negative training examples reaching the node using HOG boxes positioned using the global $p(E)$ distribution over the training set. The most discriminative ones are selected by minimizing False Positive (FP) and False Negative (FN) rates. The tree learning and the feature vector construction are depicted in the algorithm 1.

Algorithm 1: Discriminative Features Selection

input : Hierarchical structure and training images.
output: List of discriminative HOG blocks.

for *each level l* **do**
  for *each node n* **do**
    Compute edge probability $p(E_{l,n})$ over images that pass through n;
    for *each child c* **do**
      Compute edge probability $p(E_{l,n,c})$ over images that pass through c;
      Compute log-likelihood $L_{l,n,c}$ (cf Sec. 3.1);
      Sample n_h HOG blocks $\{h_i\}_{i=1}^{n_h}$ from $L_{l,n,c}$.
      for *each* h_i **do**
        Extract h_i for all the images reaching n;
        Compute the mean histogram $\bar{h_i}$ over images that pass through c ;
        Compute L_2 distances to $\bar{h_i}$;
        Compute the best threshold t_i that splits the data and minimizes FP and FN rates;
      Select the N best HOG blocks that minimizes FP and FN rates;

Features are extracted from areas of high edge probability across our training set more than areas with low probability. By using this information to sample features, the

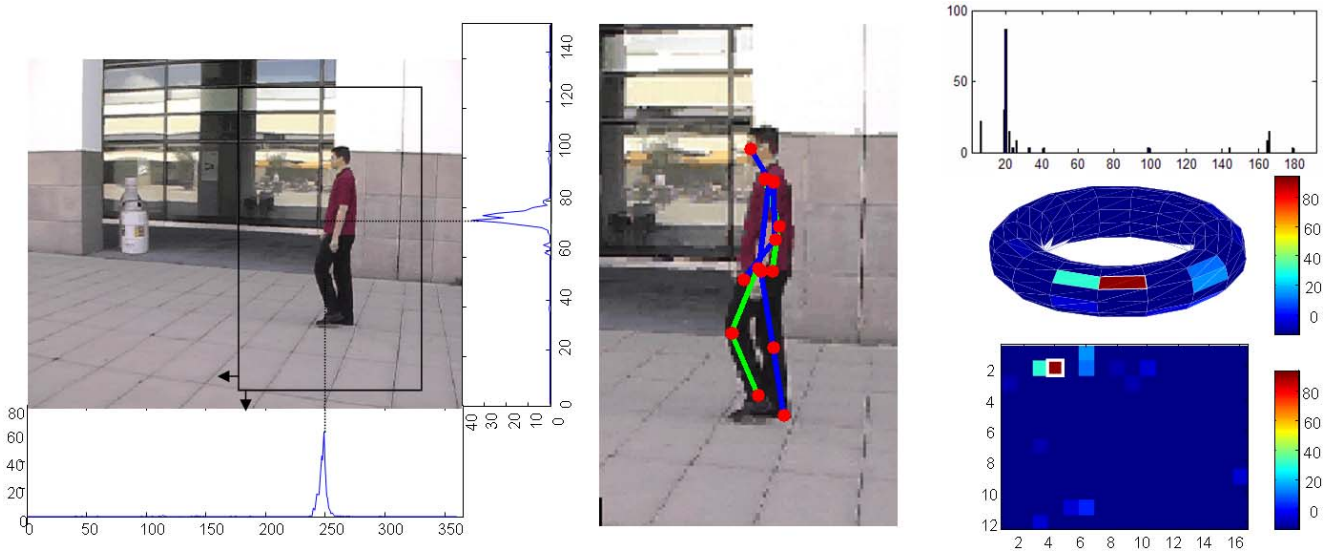


Figure 5. ***Pose Detection.*** *(a) Input image from the moving camera sequence from [11]. Scanning in X and Y directions of the image. (b) Resulting cropped image and pose corresponding to the "pick" resulting from the classification using Random Forest. (c) Resulting distribution over the 192 classes after classification using Random Forest. We also represent this distribution on the 3D and 2D representations of the torus manifold.*

amount of useful information available to the forest for random selection is increased. We then grow an ensemble of trees, a forest, by randomly sampling one of the N selected HOG blocks at each node of each tree.

4. Human Pose Detection and Classification

Given a new image, the task is to localize the individual within that image and classify the pose with the pose classifier forest. Since we learn a forest that is able to discriminate between very similar classes, we suppose that it could discriminate a human pose from the background.

Selecting and normalizing a bounding box in the input image, each tree gives a binary decision for each class, resulting in a distribution over all the classes when considering the entire forest. The decision can then be taken based on this distribution by choosing for instance the class that received more "votes". Taking the maximum classification value over the image (after exploring all the possible positions, scales and orientations) results in reasonably good localization of walking pedestrian, as shown in Fig. 5.

Once the best bounding box has been selected and classified, the 3D joints for this image are directly estimated by weighting the mean poses of the classes resulting from the distribution, using the distribution values as weights. The normalized 2D pose is computed in the same way and retransformed from the normalized bounding box to the input image coordinate system obtaining the 2D joints location.

The construction of the initial hierarchical tree structure is slightly similar to [12] even if, in our case, the process is bottom-up and the leaves are the classes we previously defined. However, it presents the following key differences with this work: first, the selection of the relevant HOG feature blocks at each node of the structure using log-likelihood ratio; and secondly, the way we grow an ensemble of trees by randomly sampling one of those selected HOG blocks at each node. This randomness makes the algorithm more robust to noise compared to a single hierarchical decision tree that would use all the features once. More importantly, it leads to a distribution over the classes that could be useful for tracking as in [21].

The algorithm used to grow the Random Forest type classifier proposed in this paper shares some similarities with PSH [19]. In PSH, the authors learn a set of hashing functions that efficiently index examples for the pose recognition task. The hash functions are sensitive to the similarity in the parameter space. In our work, we use the pose space to define classes that are then used to select relevant features and thresholds in the image space. Secondly, the hierarchical structure we build is similar to the set of projections found by PSH that best define nearest neighborhoods.

Even if our work shares some similarities with PSH, it is different in various aspects. First, thanks to our class definition, we can manage extensive viewpoint changes. Secondly, our search through the training data set is hierarchical which means that if the input vector is rejected after the first node, we do not have to extract all the other features. Finally, since the tree nodes are represented by a mean histogram and a threshold, it's possible that multiple branches can be explored as in [12] while PSH only considers binary splits. Deciding on a threshold is not an easy problem as each of the class super-sets to be separated at a given

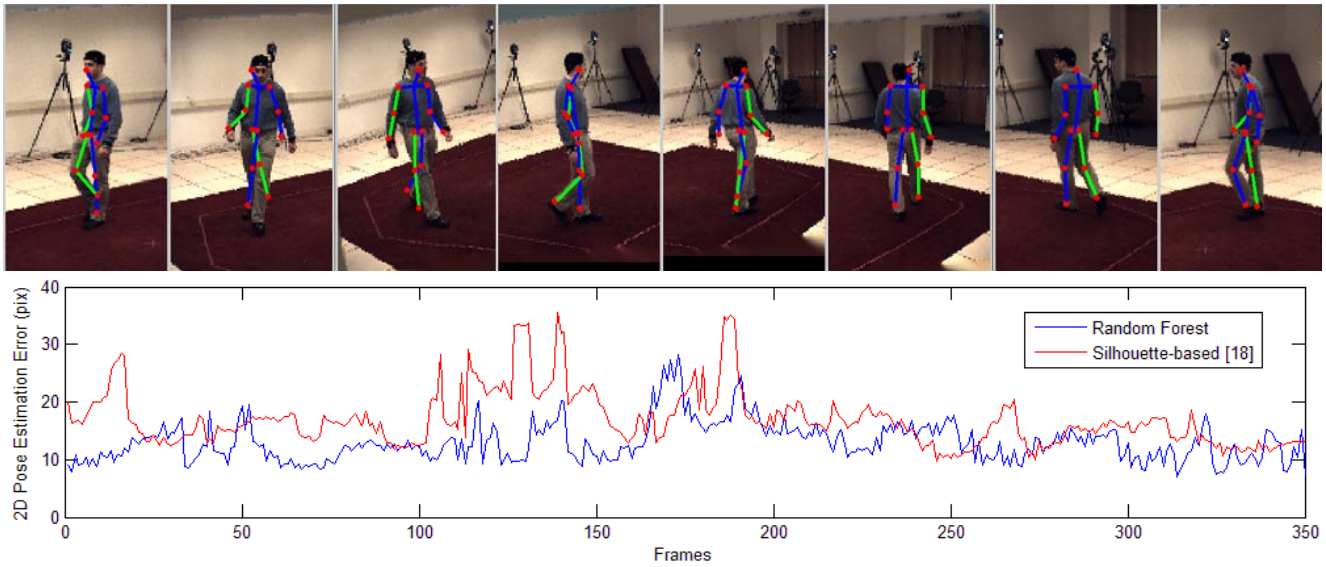


Figure 6. ***Pose detection results on HumanEva II data set.*** *Top row: normalized 96x160 images corresponding to the pick obtained when applying the pose detection in one of the HumanEva II sequences (subject S2 and Camera 1). For each presented frame (1, 50, 100, 150, 200, 250, 300 and 350) the resulting pose is represented on top of the cropped image. Bottom row: mean 2D error (in pix.) plots for the same sequence using Random Forest and [17].*

node have instances that have some similarities between each other, so exploring multiple branches is necessary.

Our approach is computationally efficient since our tree learning takes around 2 hours while SVM approaches take several days with a similar training set. Given a bounding-box, the classification with a 100 tree forest takes around 15 ms with our Matlab implementation.

5. Experiments

The proposed algorithm is first validated in similar conditions (indoor with a fixed camera) using HumanEVA II data set (Fig. 6) and secondly tested with a moving camera as in [10] (Fig. 7). We apply a simple Kalman filter on the position, scale and rotation parameters along the sequence and locally look for the maxima, selecting only the probable classes based on spatio-temporal constraints (i.e. transitions between neighbouring classes on the torus). By this process, we do not guaranty to reach the best result but a reasonably good one in relatively few iterations. Note that since we only learnt our pose detector from walking human sequences, we do not pretend detecting people performing other actions than walking. Quantitative evaluation is provided using HumanEva II data set in Tab.1.

6. Conclusions and Discussions

We have presented a novel approach to exemplar-based human pose detection and recognition using randomized trees. Unlike most previous works, our pose detection algorithm is applicable to more challenging scenarios involving extensive viewpoint changes and moving camera, without any prior assumption on an available segmented silhouette or normalized input bounding box. Moreover, our random forest classifier allows to model distribution over pose.

In future work, we will consider different actions and combine this approach with a pose tracking algorithm to make the system more robust. Finally we will improve the search to get real-time performances.

Acknowledgment

Part of this work was conducted while the first author was a visitor at Oxford Brookes University. This work was supported by spanish grant TIN2006-11044 (MEyC) and by the EPSRC research grant EP/C006631/1(P) and Sony. Gregory Rogez was funded under FPU grant AP2003-2257.

References

[1] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(1):44–58, 2006. 1, 2

[2] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Comput.*, 9(7):1545–88, 1997. 2

[3] A. Bissacco, M.-H. Yang, and S. Soatto. Detecting humans via their pose. In *NIPS*, pages 169–176, 2006. 1, 2, 4

[4] A. Bissacco, M.-H. Yang, and S. Soatto. Fast human pose estimation using appearance and motion via multi-dimensional boosting regression. In *CVPR*, 2007. 1, 2

[5] F. Bookstein. *Morphometric Tools for Landmark Data: Geometry and Biology*. Cambridge Univ. Press, 1991. 3

[6] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *ICCV*, 2007. 2

Subject	Camera	Frames	Mean (Std) from [17]	Mean (Std) using RT
S2	C1	1-350	16.96 (4.83)	12.98 (3.5)
S2	C2	1-350	18.53 (5.97)	14.18 (4.38)

Table 1. ***2D Pose Estimation Error on HumanEva II data set***. *Table showing mean (and standard deviation) of the 2D joints location error using silhouette-based model from [17] and our Randomized Trees (RT). Results are provided in pixels.*

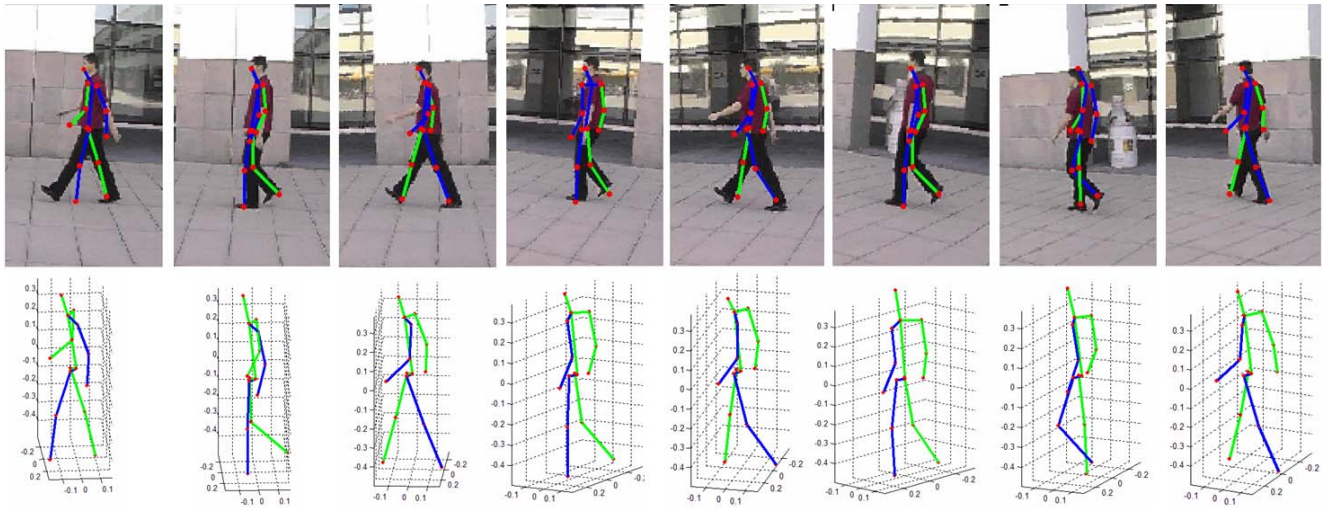


Figure 7. ***Pose detection result with a moving camera.*** *Normalized 96x160 images corresponding to the pick obtained applying the pose detection to a moving camera sequence (from [11]). For each presented frame, the mean 2D pose corresponding to the "winning" class is represented on top of the cropped image while the corresponding 3D pose is presented just below.*

[7] M. Bray, P. Kohli, and P. Torr. Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In *ECCV06*, pages II: 642–655, 2006. 1

[8] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001. 2, 4

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1, 2, 3, 4

[10] M. Dimitrijevic, V. Lepetit, and P. Fua. Human body pose detection using bayesian spatio-temporal templates. *Comput. Vis. Image Underst.*, 104(2):127–139, 2006. 1, 2, 4, 7

[11] A. Fossati, M. Dimitrijevic, V. Lepetit, and P. Fua. Bridging the gap between detection and tracking for 3d monocular video-based motion capture. In *CVPR*, 2007. 1, 6, 8

[12] D. M. Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(8):1408–1421, 2007. 1, 2, 3, 6

[13] C.-S. Lee and A. Elgammal. Simultaneous inference of view and body pose using torus manifolds. In *ICPR*, 2006. 4

[14] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Trans. PAMI*, 28(9):1465–1479, 2006. 2

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. of Comp. Vision*, 60(2):91–110, 2004. 2

[16] G. Mori and J. Malik. Recovering 3d human body configurations using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(7):1052–1062, 2006. 1

[17] G. Rogez, C. Orrite, and J. Martínez. A spatio-temporal 2d-models framework for human pose recovery in monocular sequences. *Pattern Recognition*, 2008. 1, 3, 4, 7, 8

[18] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *CVPR07*, pages 1–8, 2007. 1, 2

[19] G. Shakhnarovich, P. Viola, and R. Darrell. Fast pose estimation with parameter-sensitive hashing. In *ICCV*, 2003. 1, 2, 3, 4, 6

[20] L. Sigal and M. J. Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. 2006. 2

[21] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla. Filtering using a tree-based estimator. *ICCV*, 02, 2003. 2, 6

[22] A. Thayananthan, R. Navaratnam, B. Stenger, P. H. S. Torr, and R. Cipolla. Multivariate relevance vector machines for tracking. In *ECCV (3)*, pages 124–138, 2006. 1

[23] K. Toyama and A. Blake. Probabilistic tracking with exemplars in a metric space. *IJCV*, 48(1):9–19, 2002. 2

[24] R. Urtasun, D. Fleet, and P. Fua. Temporal motion models for monocular and multiview 3d human body tracking. *Comp. Vision Image Underst.*, 103(2-3):157–177, 2006. 1, 4

[25] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *Int. J. Comput. Vision*, 63(2):153–161, 2005. 1, 2

[26] Y. Wu, J. Lin, and T. Huang. Analyzing and capturing articulated hand motion in image sequences. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 27(12):1910–1922, 2005. 1, 2

[27] Q. Zhu, M. Yeh, K. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR06*, pages II: 1491–1498, 2006. 1, 2, 4