Fast Human Pose Estimation using Appearance and Motion via Multi-Dimensional Boosting Regression

Alessandro Bissacco
Google, Inc.
605 Arizona Avenue
Santa Monica, CA 90401
bissacco@gmail.com

Ming-Hsuan Yang
Honda Research Institute
800 California Street
Mountain View, CA 94041
mhyang@ieee.org

Stefano Soatto
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095
soatto@cs.ucla.edu

Abstract

We address the problem of estimating human pose in video sequences, where rough location has been determined. We exploit both appearance and motion information by defining suitable features of an image and its temporal neighbors, and learning a regression map to the parameters of a model of the human body using boosting techniques. Our algorithm can be viewed as a fast initialization step for human body trackers, or as a tracker itself. We extend gradient boosting techniques to learn a multi-dimensional map from (rotated and scaled) Haar features to the entire set of joint angles representing the full body pose. We test our approach by learning a map from image patches to body joint angles from synchronized video and motion capture walking data. We show how our technique enables learning an efficient real-time pose estimator, validated on publicly available datasets.

1. Introduction

An important problem in modern computer vision is full body tracking of humans in video sequences. In this work we focus in particular on estimating the 3D pose of a kinematic model of the human body from images. Such a task is extremely challenging for several reasons. First there exist multiple plausible solutions to a query, since we are trying to recover 3D information from 2D images (this is especially true in the presence of partial occlusions). In order to disambiguate such cases, we can use prior knowledge on the most likely configurations, for example in a walking gait we expect the occluded arm to be parallel to the torso.

Second, humans are articulated objects with many parts whose shape and appearance change due to various nuisance factors such as illumination, clothing, viewpoint and pose. This fact causes difficulties when using a discriminative approach (e.g. [19]) to learn the map from images to poses, or when using a generative approach (e.g. [5]) to build a likelihood function as a matching score between a configuration hypothesis and a given image. Consequently, it is common to extract a feature representation which is insensitive to nuisance factors. For pose estimation, a frequent choice is binary silhouettes, which can be computed from images using motion, a background model, or a combination of the two [1, 15, 7]. Using only silhouettes is limiting, since important appearance information is discarded, which could help resolving ambiguous cases.

Finally, the space of admissible solutions, that is all possible positions and orientations of all body parts, is extremely large, and the search for the optimal configuration in this space is a combinatorial problem. To address this issue, most approaches proposed so far attempt to reduce the feasible space using both static and dynamic constraints. Static constraints restrict the search to the set of physically feasible body configurations. Dynamic constraints work by enforcing temporal continuity between adjacent frames, specified through a set of motions. A common approach is to learn a statistical model of the human dynamics and to use it in a sampling scheme where, given the body configuration in the current frame and the motion model, we can compute a probability distribution which allows us to make informed guesses on the limb positions in the next frame.

Although learned motion models have been shown to greatly improve tracking performance for simple motions such as walking gaits, it is not clear how to efficiently combine different models in order to represent the ample variety of motions that can be performed by humans. Indeed, in the literature, examples of effective tracking are limited to a small number of motions not too different from the training dataset. Moreover, each learned model represents a particular motion at a particular speed, so the system is unlikely to successfully track even an instance of the same motion if performed at a speed different from the one used for learning.

In general, there are conditions where the tracker either provides an inaccurate estimate or loses track altogether.

1-4244-1180-7/07/$25.00 ©2007 IEEE

This is particularly true for fast motions, where the body limbs undergo large displacements from one frame to the next. Recent approaches which have shown considerable success for fast motions perform tracking by doing pose estimation independently at each frame [13]. Although we do not argue that this is necessarily the right approach to tracking, we believe in the importance of having an efficient pose estimator, which can take action whenever the tracking algorithm fails. Therefore, the focus of this work is on building a fast body pose estimator for human tracking applications. Our pose estimator can be applied for automatically initializing a tracking module in the first frame and reinitializing it every time it loses track, or by running it at every frame, as a tracking algorithm.

The main distinction of our approach with respect to current state-of-the-art human pose estimators is that we aim to develop an algorithm which is fast enough to be run at every frame and used for real-time tracking applications. Unavoidably, to accomplish this we have to trade-off estimation accuracy for execution speed.

Our work can also be seen as an element of an effective automatic body pose estimator system from video sequences. On one hand we have efficient body detectors [22] which can estimate presence and location of people in images. On the other hand we have accurate but computationally expensive dynamic programming approaches [5] which can find the optimal pose estimate of an articulated body model in a neighborhood of a proposed body configuration. Our method bridges the gap between these two approaches by taking an image patch putatively containing a human and computing an initial guess of her body pose, which can be later refined using one of the pose estimators available in the literature.

An important characteristic of our approach is that, in order to estimate the body pose, instead of restricting to binary silhouettes, we exploit both appearance and motion. By doing so we can resolve some of the ambiguities that we would face if trying to directly map silhouettes to poses and which have led many researchers in this field to employ sophisticated mixture models [2, 19, 20].

2. Related work

Estimating pose from a single image without any prior knowledge is an extremely challenging problem. It has been cast as deterministic optimization [5, 14], as inference over a generative model [9, 11, 8, 18], as segmentation and grouping of image regions [12], or as a sampling problem [9]. Proposed solutions either assume very restrictive appearance models [5] or make use of cues, such as skin color [23] and face position [11], which are not reliable and can be found only in specific classes of images (e.g. sport players or athletes).

A large body of work in pose estimation focus on the simpler problem of estimating the 3D pose from human body silhouettes [1, 15, 19, 7]. It is possible to learn a map from silhouettes to poses, either direct [1], one-to-many [15] or as a probabilistic mixture [2, 19]. However, as we mentioned in the introduction, silhouettes are inherently ambiguous as very different poses can generate similar silhouettes, so to obtain good results either we resort to complex mixture models [19] or restrict the set of poses [3], or use multiple views [7]. Shakhnarovich et al. [16] demonstrates that combining appearance with silhouette information greatly improves the quality of the estimates. Assuming segmented images, they propose a fast hashing function that allows matching edge orientation histograms to a large set of synthetic examples. We experimented with a similar basic representation of the body appearance, by masking out the background and computing our set of oriented filters on the resulting patch.

Besides silhouettes and appearance, motion is another important cue that can be used for pose estimation and tracking [4, 24]. Most works assume a parametric model of the optical flow, which can be either designed [24] or learned from examples [4]. But complex motion models are not the only way to make use of motion information. As shown in [22], simple image differences can provide an effective cue for pedestrian detection. We follow this path, and integrate our representation of human body appearance with motion information from image differences.

Finally, recent work [13] advocates tracking by independently estimating pose at every frame. Our approach has a natural application in such a scenario, given that it can provide estimates in remarkably short order and, unlike [13], one does not need to learn an appearance model specific to a particular sequence.

3. Appearance and Motion Features for Pose Estimation

The input to our algorithm is a video sequence, together with the bounding boxes of the human body for each frame as extracted by a detector (e.g. [22]). We do not require continuity of the detector responses across frames, however our approach cannot provide an estimate for the frames in which the human body is not detected. If available, our approach may also take advantage of the binary silhouettes of the person, which can be extracted from the sequence using any background subtraction or segmentation scheme. However, in practical real-time scenarios the quality of the extracted silhouettes is generally low and in our experiments we noticed that using bad silhouettes degrades the estimator performance.

In this section we introduce our basic representation of appearance and motion for the pose estimation problem. We use a set of differential filters tailored to the human body to extract essential temporal and spatial information from the

images. We create a large pool of features, which later will be used in a boosting scheme to learn a direct map from image frames to 3D joint angles.

3.1. Motion and Appearance Patches

The starting point of our algorithm are patches containing the human body, extracted from the image frames using the bounding boxes provided by a human body detector. Patches are normalized in intensity value and scaled to a default resolution (64×64 in our experiments).

We can use the silhouette of the human body (extracted by any background subtraction technique) to mask out the background pixel in order to improve learning speed and generalization performance. However, this step is by no means necessary: Given sufficient amount of data and training time, the boosting process automatically selects only the features whose support lies mostly in the foreground region. In our experiments we noticed that using low quality silhouettes compromises performance, so we opted to omit this preprocessing step.

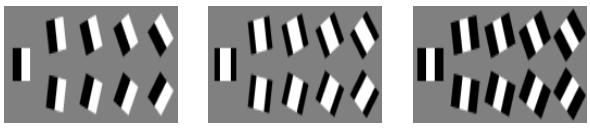
Motion information is represented using the absolute difference of image values between adjacent frames: $\Delta_i = \text{abs}(I_i - I_{i+1})$. As done before, from the image difference Δ_i we compute the motion patches by extracting the detected patch. We could use the direction of motion as in [22] by taking the difference of the first image with a shifted version of the second, but in order to limit the number of features considered in the training stage we opted for not using this additional source of information. In Figure 2 we can see some sample appearance and motion patches.

Normalized appearance I_i and motion Δ_i patches together form the vector input to our regression function: $\mathbf{x_i} = \{I_i, \Delta_i\}$.

3.2. Features for Body Parts

Our human pose estimator is based on Haar-like features similar to the ones proposed by Viola and Jones in [22]. These filters measure the difference between rectangular areas in the image with any size, position and aspect ratio. They can be computed very efficiently from the integral image. However, in the context of this work a straightforward application of these filters to appearance and motion patches is not doable for computational reasons.

For detection of either faces or pedestrians, a small patch of about 20 pixels per side is enough for discriminating the object from the background. But our goal is to extract full pose information, and if we were to use similar resolutions we would have limbs with area of only a few pixels. This would cause their appearance to be very sensitive to noise and would make it extremely difficult to estimate pose. We chose the patch size by visual inspection, perceptually determining that a 64×64 image contains enough information for pose estimation by a human observer. Unfortunately, augmenting the patch size greatly increases the number of basic features that fit in the patch (approximately squared in its area), therefore we need a strategy for selecting a good subset for training.



(a) (b) (c)

Figure 1. Basic types of Haar features used in this work: edges (a), thick (b) and thin (c) lines. Each of these features can assume any position and scale within the estimation window (although for scale some restrictions apply, see text for details). Each feature can assume a set of 18 equally spaced orientations in the range $[0, \pi]$, here we show the 9 horizontal orientations, vertical ones are obtained by swapping axes. The value of the feature is computed by subtracting the sum of pixels values inside white regions from pixels in black regions, scaled by their area. It is intuitive to see how features (c) are suitable to match body limbs, while features (a) and (b) can be used to match trunk, head and full body.

Another weakness of these basic features is that, by using vertical rectangles only, they are not suited to capture edges that are not parallel to the image axes. For pose estimation this is a serious shortcoming, since the goal is to localize limbs which can have arbitrary orientation. Therefore, we extended the set of basic Haar features by introducing their rotated versions, computed at a few major orientations, as shown in Figure 1. Notice that these filters are very similar to oriented rectangular templates commonly used for detecting limbs in pose detection approaches [5, 13]. Oriented features can be extracted very efficiently from integral images computed on rotated versions of the image patch. Notice that by introducing orientation in the features we further increase their number, so a good subset selection in the training process becomes crucial.

We experimented with various schemes for feature selection. Among the possible configurations, we found that one type of edge feature (Figure 1a) and two types of lines features (Figure 1b and 1c) are the best performers. Each feature can assume any of 18 equally spaced orientations in the range $[0, \pi]$, and they can have any position inside the patch. To limit the number of candidates, we restrict each rectangle to have a minimum area of 80 pixels, do not come closer than 8 pixels from the border, have even width and even height.

With this configuration, we obtain a pool of about 3 million filters for each of the motion and image patches. Since this number is still too high, we randomly select K of these features by uniform sampling. The result is a set of features $\{f^k(\mathbf{x}_i)\}_{k=1,\cdots,K}$ that map motion and appearance patches $\mathbf{x}_i = \{I_i, \Delta_i\}$ to real values.

4. Multidimensional Gradient Boosting

In this section we introduce a novel approach for learning the regression map from motion and appearance features to 3D body pose.

We start with the robust boosting approach to regression proposed in [6]. This algorithm is particularly suited to our problem since it provides an efficient way to automatically select from the large pool of filters the most informative ones to be used as basic elements for building the regression function. Our contribution is to extend the gradient boosting technique [6] to multidimensional maps. Instead of learning a separate regressor for each joint angle, we learn a vector function from features to sets of joint angles representing full body poses.

The advantage of learning multidimensional maps is that it allows the joint angle estimators to share the same set of features. This is beneficial because of the high degree of correlation between joint angles for natural human poses. The resulting pose estimator is sensibly faster than the collection of scalar counterparts, since it uses a number of features which grows with the effective dimension of the target space instead of with the number of joint angles. This has some similarities with the work of Torralba et al. [21], where detectors of a multiclass object classifier are trained jointly so that they share set of features.

An approach closely related to ours is the multidimensional boosting regression of Zhou et al. [25]. There, the regression maps are linear combinations of binary functions of Haar features, with the additional constraint that all regressors have the same coefficients. Restricting the learned maps to such a simple function class allows the authors to derive a boosting-type gradient descent algorithm that minimizes the least-squares approximation error in closed-form. However, such a representation is not suited to fit multidimensional maps having components at different scales, it cannot be easily extended to include more complex basic functions such as regression trees, and most importantly there is no sharing of features between regressors. We propose an approach that overcomes these limitations and can successfully learn maps from image patches to 3D body pose.

In the next section we review the basic gradient boosting algorithm, then we derive our extension to multidimensional mappings.

4.1. Gradient Treeboost

Given a training set $\{y_i, \mathbf{x}_i\}_1^N$, with inputs $\mathbf{x}_i \in \mathbb{R}^n$ and outputs $y_i \in \mathbb{R}$ independent samples from some underlying joint distribution, the goal of regression is to find a function $F^*(\mathbf{x})$ that maps $\mathbf{x}$ to y, such that the expected value of a loss function $E_{\mathbf{x},y}\left[\Psi(y, F(\mathbf{x}))\right]$ is minimized. Typically, the expected loss is approximated by its empirical estimate, thus the regression problem can be written as:

$$F^*(\mathbf{x}) = \arg\min_{F(\mathbf{x})} \sum_{i=1}^{N} \Psi(y_i, F(\mathbf{x}_i)). \tag{1}$$

In this work we impose regularization by assuming an additive expansion for $F(\mathbf{x})$ with basic functions h:

$$F(\mathbf{x}) = \sum_{m=0}^{M} h(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m) \tag{2}$$

where $h(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m) = \sum_{l=1}^{L} a_{lm} 1(\mathbf{x} \in R_{lm})$ are piecewise constant functions of $\mathbf{x}$ with values $\mathcal{A}_m = \{a_{1m}, \cdots, a_{Lm}\}$ and input space partition $\mathcal{R}_m = \{R_{1m}, \cdots, R_{Lm}\}$[1]. For $L = 2$ our basic functions are decision stumps, which assume one of two values according to the response of a feature $f^{k_m}(\mathbf{x})$ compared to a given threshold θ_m. In general h is a L-terminal node Classification and Regression Tree (CART)[10], where each internal node splits the partition associated to the parent node by comparing a feature response to a threshold, and the leaves describe the final values $\mathcal{A}_m$.

We solve (1) by a greedy stagewise approach where at each step m we find the parameters of the basic learner $h(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m)$ that maximally decreases the loss function (1):

$$\mathcal{A}_m, \mathcal{R}_m = \underset{\mathcal{A},\mathcal{R}}{\operatorname{argmin}} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\mathbf{x}_i) + h(\mathbf{x}_i; \mathcal{A}, \mathcal{R})) \tag{3}$$

Since the basic learner is a piecewise-constant function, solving (3) by gradient descent on the parameters is infeasible - it is easy to see that the partial derivatives of h with respect to R_{im} are Dirac deltas.

We apply Gradient Treeboost [6], an efficient approximate minimization scheme solving (1) with a two-step approach. At each stage m it uses the previous estimate F_{m-1} to compute the "pseudo-residuals" $\tilde{y}_{im} = -\left[\frac{\partial \Psi(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}$

First it finds the input space partition $\mathcal{R}_m$ (a L-node regression tree) by least-squares fitting the basic learner $h(\mathbf{x}; \mathcal{A}, \mathcal{R})$ to the pseudo residuals:

$$\tilde{\mathcal{A}}_m, \mathcal{R}_m = \underset{\mathcal{A},\mathcal{R}}{\operatorname{argmin}} \sum_{i=1}^{N} |\tilde{y}_{im} - h(\mathbf{x_i}; \mathcal{A}, \mathcal{R})|^2 \tag{4}$$

When the basic learners h are decision stumps constructed from a pool of K features, the solution to (4) is found by estimating for each feature f^{k_m} the threshold θ_m and approximating values a_{1m}, a_{2m} minimizing (4), and picking the one with the lowest error. This step is equivalent to solving (3) assuming least-squares loss $\Psi(y, x) =$

[1] Here we denote $1(c)$ the function that is 1 if condition c is true, 0 otherwise.

$|y-x|^2$. Then it computes the regression tree values $\mathcal{A}_m$ by optimizing the original loss function $\Psi(y, F(\mathbf{x}))$ within each partition R_{lm}, i.e. by finding the constant offset a_{lm} to the previous approximation F_{m-1} that best fits the measurements:

$$a_{lm} = \underset{a}{\operatorname{argmin}} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\mathbf{x_i}) + a) 1(\mathbf{x_i} \in R_{lm}). \quad (5)$$

The pseudo residuals $\tilde{y}_{im}$ and the tree predictions a_{lm} depend on the choice of the loss criterion Ψ.

In the case of Least Squares (LS) $\Psi(y, F(\mathbf{x})) = |y - F(\mathbf{x})|^2$, the pseudo residuals are just the current residuals:

$$\tilde{y}_{im} = y_i - F_{m-1}(\mathbf{x_i}) \quad (6)$$

and both input partition $\mathcal{R}$ and function values $\mathcal{A}$ are computed in the first step (4). In this case the Gradient TreeBoost algorithm reduces to (3).

Using Least-Absolute-Deviation (LAD or L_1 error) $\Psi(y, F(\mathbf{x})) = |y - F(\mathbf{x})|$, we have:

$$\begin{aligned} \tilde{y}_{im} &= \text{sign}(y_i - F_{m-1}(\mathbf{x_i})) \\ a_{lm} &= \text{median}_{i:\mathbf{x_i} \in R_{lm}} \{y_i - F_{m-1}(\mathbf{x_i})\}. \end{aligned} \quad (7)$$

An important feature of the Gradient TreeBoost algorithm is that, before updating the current approximation, the estimated regression tree is scaled by a shrinkage parameter $0 < \nu < 1$, where ν controls the learning rate (smaller values lead to better generalization):

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu \sum_{l=1}^{L} a_{lm} 1(\mathbf{x} \in R_{lm}). \quad (8)$$

In our setting, the regions are defined by thresholds θ on filter responses $f^k(\mathbf{x})$, where f^k is the k-th Haar filter computed on the appearance and motion patches $\mathbf{x} = \{I, \Delta\}$. For the case of degenerate regression trees with a single node (decision stumps), we have:

$$h_s(\mathbf{x}; a_{1m}, a_{2m}, k_m, \theta_m) = \begin{cases} a_{1m} & \text{if} \quad f^{k_m}(\mathbf{x}) \leq \theta_m \\ a_{2m} & \text{if} \quad f^{k_m}(\mathbf{x}) > \theta_m \end{cases} \quad (9)$$

Notice that these basic learners are more general than the ones proposed in [25], since we do not have the constraint that $a_{2m} = -a_{1m}$. Additionally, while [25] is restricted to decision stumps as basic functions, our boosting framework supports general regression trees. As we show in the experiments (Figure **4**), boosting Classification and Regression Trees yields regressors clearly having higher accuracy than boosted decision stumps.

4.2. Multidimensional Gradient TreeBoost

In this section we propose an extension to the Gradient TreeBoost algorithm in order to efficiently handle multidimensional maps.

Given a training set $\{\mathbf{y}_i, \mathbf{x}_i\}_1^N$ with vector inputs $\mathbf{x}_i \in \mathbb{R}^n$ and outputs $\mathbf{y}_i \in \mathbb{R}^p$, our goal is to find the map $\mathbf{F}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^p$ minimizing the loss $\Psi(\mathbf{y}, \mathbf{F}(\mathbf{x}))$.

As in the previous section, Multidimensional Treeboost is derived by assuming that the map $\mathbf{F}(\mathbf{x})$ can be expressed as a sum of basic piecewise constant (vector) functions:

$$\mathbf{F}(\mathbf{x}) = \sum_{m=0}^{M} \mathbf{h}(\mathbf{x}; \{\mathcal{A}_m^1, \cdots, \mathcal{A}_m^p\}, \mathcal{R}_m) = \begin{bmatrix} \sum_{m=0}^{M} h(\mathbf{x}; \mathcal{A}_m^1, \mathcal{R}_m) \\ \cdots \\ \sum_{m=0}^{M} h(\mathbf{x}; \mathcal{A}_m^p, \mathcal{R}_m) \end{bmatrix} \quad (10)$$

and by minimizing $E_{\mathbf{y},\mathbf{x}} \Psi(\mathbf{y}, \mathbf{F}(\mathbf{x}))$ using the Gradient Treeboost scheme described in the previous section.

Notice that (10) differs from applying the expansion (2) to each element of the vector map $\mathbf{F}(\mathbf{x})$ in that we restrict all the basic functions $h_i(\mathbf{x}) = h(\mathbf{x}; \mathcal{A}^i, \mathcal{R}^i)$ to share the same input space partition: $\mathcal{R}^i \equiv \mathcal{R}$. For our application, this translates into requiring all the joint angle regressors to share the same set of features, thereby substantially improving the efficiency of the representation.

Let us also point out the main difference with respect to the multidimensional boosting regression of Zhou et al. [25]. There, correlation between regressors is enforced by restricting the basic functions to have the same absolute value at each step, i.e. $|a_{1m}^i| = |a_{2m}^i| \equiv a_m, i \in \{1, \cdots, p\}$. Such modeling assumption allows us to solve (3) for least-squares loss with an efficient gradient based approach. However, it is clear that such a representation can effectively describe only multidimensional processes with equally scaled components, so a whitening preprocessing step is required. Also, using a least-squares loss does not provide robustness to outliers. Most importantly, we obtain a different set of features for each output component i, and for high-dimensional output spaces this yields inefficiency in the learned maps.

Using decision stumps on Haar feature responses as basic learners and assuming Least Squares or Least Absolute Deviation loss functions we obtain the simple versions of Multidimensional Gradient Treeboost shown in Algorithm 1. Here we give a brief outline of the main steps of the algorithm.

The approximation is initialized in line 1 with the constant function minimizing the loss function, i.e. either the mean or the median of the training outputs $\mathbf{y}_i$ depending on the loss function. At line 3 the pseudo-residual vectors are computed, as either the current training residuals $\mathbf{y}_i - F(\mathbf{x}_i)$ or their signs. Line 4 computes the regions R_{lm} by finding optimal feature and associated threshold value: For every feature f^k, we compute the least-squares approximation er-

Algorithm 1 Multidimensional Gradient TreeBoost for Least-Squares (LS) and Least-Absolute-Deviation (LAD) loss.

1: $\mathbf{F}_0(\mathbf{x}) = \begin{cases} \text{mean}\{\mathbf{y_i}\}_{i=1,\cdots,N} & \textbf{LS} \\ \text{median}\{\mathbf{y_i}\}_{i=1,\cdots,N} & \textbf{LAD} \end{cases}$

2: **for** $m = 1$ to M **do**

3: $\tilde{\mathbf{y}}_{im} = (\tilde{y}^1_{im}, \cdots, \tilde{y}^p_{im}) = \begin{cases} \mathbf{y}_i - \mathbf{F}_{m-1}(\mathbf{x}_i) \quad , \quad i = 1, \cdots, N & \textbf{LS} \\ \tilde{\mathbf{y}}_{im} = \text{sign}\left(\mathbf{y}_i - \mathbf{F}_{m-1}(\mathbf{x}_i)\right) & \textbf{LAD} \end{cases}$

4: $k_m, \theta_m = \arg\min_{k,\theta} \sum_{j=1}^{p} \min_{a_1,a_2} \sum_{i=1}^{N} \left(\tilde{y}^j_{im} - h_s(\mathbf{x}_i; a_1, a_2, k, \theta)\right)^2$

5: $\mathbf{a}_{1m}, \mathbf{a}_{2m} = \begin{cases} \text{mean}\{\mathbf{y_i} - \mathbf{F}_{m-1}(\mathbf{x}_i)\}_{i:f^k(\mathbf{x}_i)<\theta} \quad , \quad \text{mean}\{\mathbf{y_i} - \mathbf{F}_{m-1}(\mathbf{x}_i)\}_{i:f^k(\mathbf{x}_i)\geq\theta} & \textbf{LS} \\ \text{median}\{\mathbf{y_i} - \mathbf{F}_{m-1}(\mathbf{x}_i)\}_{i:f^k(\mathbf{x}_i)<\theta} \quad , \quad \text{median}\{\mathbf{y_i} - \mathbf{F}_{m-1}(\mathbf{x}_i)\}_{i:f^k(\mathbf{x}_i)<\theta} & \textbf{LAD} \end{cases}$

6: $\mathbf{F}_m(\mathbf{x}) = \mathbf{F}_{m-1}(\mathbf{x}) + \nu \mathbf{h}_s\left(\mathbf{x}; \mathbf{a}_{1m}, \mathbf{a}_{2m}, k_m, \theta_m\right)$

7: **end for**

rors to the pseudo-residuals $\tilde{\mathbf{y}}_{im}$ using p vector stumps h_s whose inputs are the filter responses $f^k(\mathbf{x}_i)$, and pick the one with the lowest error. This is the most computationally expensive part of the algorithm, since we need to evaluate all features $k = 1, \cdots, K$ on the entire training set. Notice that the least-squares criterion allows us to efficiently find the values a_i, since we only need to incrementally compute the mean of the outputs sorted by feature value while we search for the optimal threshold. Line 5 finds the two vector parameters $\mathbf{a}_1, \mathbf{a}_2$ of the basic stump learner $\mathbf{h}_s$, which are the constant predictions of the residuals in the two regions found in the previous step. For least-squares they are the values a_i computed in step 4, for L_1 loss $\mathbf{a}_j$ they are the medians of the sample residuals. Line 6 adds the stump classifier $\mathbf{h}_s$ to the current vector function approximation $\mathbf{F}_m$, scaled by the learning rate ν.

As the name suggests, this algorithm is not limited to stumps but can be formulated for arbitrary decision trees. For the sake of clarity, we presented here only the simplest case. However, in the experiments we also show results from applying Classification and Regression Trees (CART) [10] as basic functions $\mathbf{h}(\mathbf{x})$. These are decision trees modeling a piecewise constant function, where each node of the tree uses a feature f^k and a threshold θ to recursively split the current region of the input space in two, and the terminal leaves define the input space partition R_{lm}.

5. Experiments

In our experiments we used the synchronized video and human motion dataset for human tracking and pose estimation recently made publicly available by the Brown Group [17]. The dataset consists of 4 views of people with motion capture makers attached to their body walking in a circle, see Figure 2 for some sample frames. In our experiments we use only the walking sequences for which both video and motion data are available, having a total of three subjects and 2950 frames (first trial of subjects S1, S2, S3). Our goal is pose estimation from a single view, therefore we use only the images taken from a single camera (C1). In order to assess the performance of our pose estimator and compare it with alternative approaches to boosting regression we trained the model using 5-fold cross-validation.

Motion information for this data consists in the 3D transformations from a global reference frame to the body part local coordinates. We have a total of 10 parts (head, torso, upper and lower arms, upper and lower legs). We represent motion as the relative orientation of adjacent body parts expressed in the exponential map coordinates. By discarding coordinates that have constant value in the performed motions we reduce to 26 degrees of freedom.

The first step of our approach is to extract the human body patches and scale them to the default resolution of 64×64 pixels and normalized in intensity. It may be helpful to mask out the background pixels from the patches using binary silhouettes extracted by background subtraction. However, in our experiments such a preprocessing step actually degrades performance, the reason may be the low quality of the silhouettes employed. We then extract motion patches as the differences between adjacent frames, scaled and normalized as just described. Although eventually in real applications the patches will be provided by a detector, we used the calibration information available in the datasets to draw the patches. Some sample output of this preprocessing stage are reported in Figure 3.

To facilitate comparison with Zhou et al. [25], we normalize the joint angle trajectories, so that $\mathbf{y}$ is a zero-mean unit-variance process. In this way each joint angle contributes equally to the cost function (1), while better visual results could be obtained by scaling the joint angles according to their contribution to the final image.

Given a set of motion and appearance patches $\mathbf{x}_i$ with associated normalized joint angles $\mathbf{y}_i$, we use our approach to train a multidimensional boosting regressor for both least-squares and least-absolute-deviation loss function, using either decision stumps (Algorithm 1) or CART as basic learners, and compare with the results from our implementation of [25]. At each iteration, for each patch we evaluated 10^5 features randomly sampled from the pool of oriented filters described in section 3.2. We experimentally found the optimal learning rate $\nu = 0.5$, and run the boosting process until the improvement on the training residual is negligible. We also experimented with different basic functions, and ob-

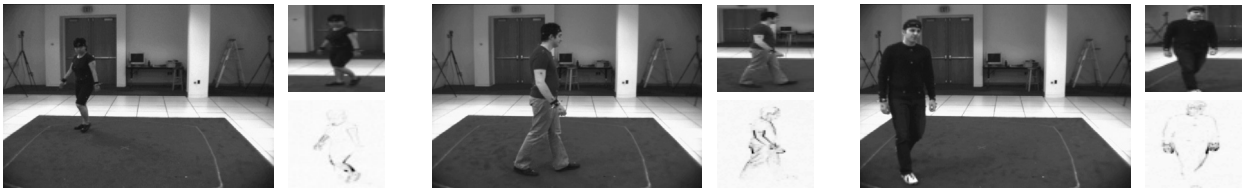


Figure 2. Sample frames from the dataset, and extracted appearance and motion patches. There are 3 subjects (S1, S2, S3), each performing a pair of cycles in a circular walking motion, for a total of 2950 frames (1180, 875 and 895 respectively for each subject). In the first row we show a sample frame for each subject. In the second row, we display appearance and motion patches, scaled to the default resolution 64×64 and normalized in intensity.

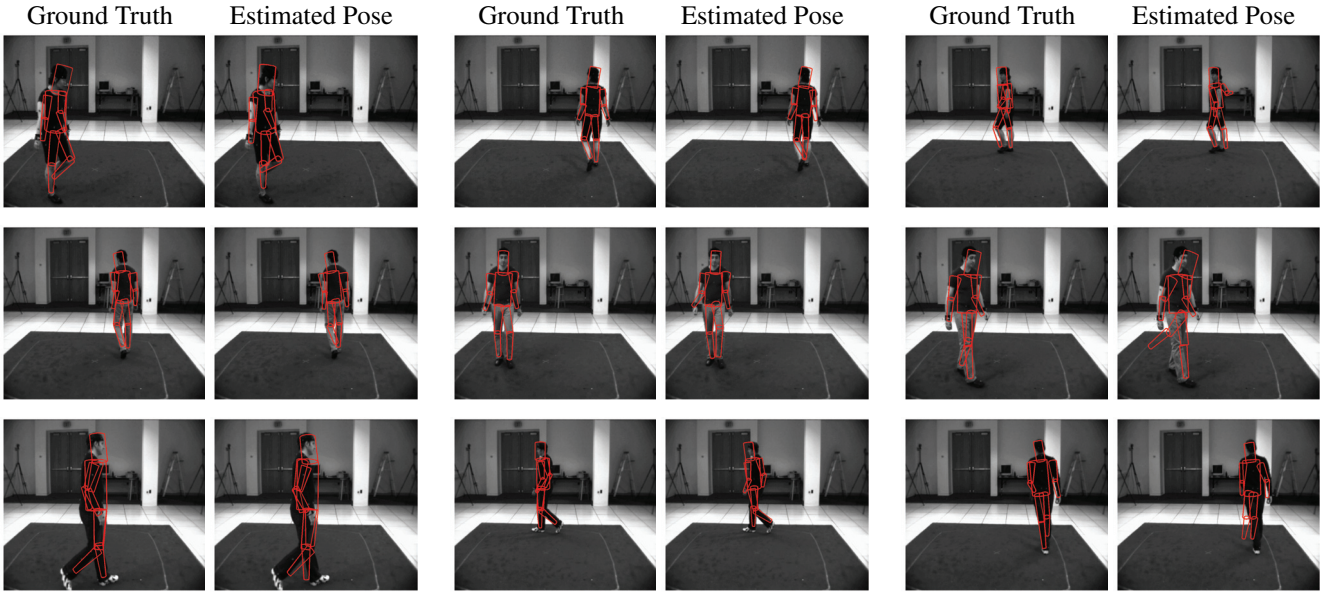


Figure 3. Sample estimation results. First, third and fifth columns show the provided ground truth, while second, forth and last column show the estimated pose. Samples in the last column show that, as we could expect, large estimation errors occur for poses characterized by prominent self-occlusions.

tained best results using 5-node Classification and Regression Trees. We believe that decision stumps do not perform as well for this kind of problem because body part configurations for articulated objects such as humans are highly dependent, and an approximation of the pose map (10) as a sum of functions of single features cannot capture these dependencies. On the other hand, CART trees of n nodes can model functions having arbitrary interactions between $n-1$ variables. Figure 4 shows mean and standard deviation of the L_1 and L_2 pose error norms on the entire dataset during validation, together with a plot of the errors on the individual joint angles. Our approach outperforms [25] in estimation accuracy. Rather surprisingly, the least-squares approach outperforms the least-absolute-deviation in all settings. Since we noticed similar results with other motion representations (i.e. 2D marker positions), we believe that this is due to the approximated nature of the algorithm for non-squared loss, and for a dataset with few outliers the benefits of the L_1 criterion are overcome by the error in the approximation. We also report the number of features used by the trained regressor, that shows the advantage of our multidimensional approach in selecting a smaller number of features over alternative approaches [25]. Besides computational speed (about 1.2 milliseconds per patch for our Least-Squares CART regressor), notice the memory efficiency of our approach, given that each basic functions is represented with the descriptions of the Haar features (orientation and location of the rectangles), thresholds and output values. Compare this with approaches based on exemplar matching or kernel machines [2, 20], which often need to retain a large part of the training examples. In Figure 3 we show some sample motion and appearance patches together with the estimated pose represented as the outline of a cylinder-based human model superimposed onto the original images. Here we use a regressor trained using least-absolute-deviation and 5-node CARTs as basic functions. From these results it is clear that the lack of prior information adversely affects the estimations of occluded parts.

6. Conclusions

In this work we proposed a novel approach to estimate 3D human poses from images. We derived an efficient algorithm which can run in real time and extract full 3D body

Algorithm	Mean	Standard Deviation	Number of Features	Time (s)
Zhou et al. [25]	0.3031 (0.3026)	0.0750 (0.0703)	52000	40.55
LS stump	0.2818 (0.2777)	0.0931 (0.0885)	2000	4.22
LAD stump	0.2922 (0.2757)	0.1020 (0.0909)	8000	16.14
LS 5-CART	**0.2736** (**0.2701**)	**0.1158** (**0.1067**)	**2850***	**3.28**
LAD 5-CART	0.2996 (0.2863)	0.0972 (0.0929)	6000*	5.94

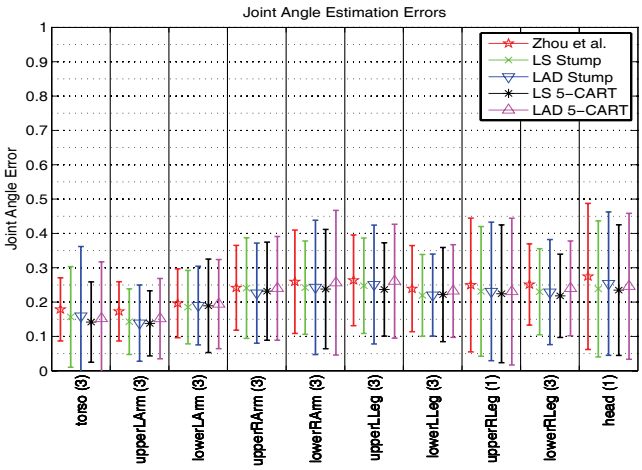


Figure 4. Pose Estimation Errors. (Left) Table showing mean and standard deviation of the relative L_2 error norm (i.e. $||\hat{\mathbf{y}}_i - \mathbf{y}_i||/||\mathbf{y}_i||$) for the entire dataset during validation phase. In brackets are the mean and standard deviation of relative L_1 error norm. We report results for the multidimensional boosting [25] and our Multidimensional TreeBoost with both Least Squares (LS) and Least Absolute Deviation (LAD) loss, using stumps and Classification and Regression Trees [10] with 5 internal nodes as basic functions. We see that the best performer (in bold) is the CART regressor with Least-Squares loss, while the approaches using LAD criterion do not score as well. For each classifier we also report the number of features used (for CART regressors the * denotes an upper bound, for each evaluation only a subset of features are computed) and the evaluation time on the entire dataset (2950 frames): The efficiency of our approach is evident. (Right) Plot of mean values and standard deviations of the joint angle relative errors for each limb, in parenthesis the number of degrees of freedom of the parts. From the plot we see that the highest of the errors concentrates on the limbs, since they are more prone to occlusions.

pose estimates from image patches containing humans. We introduced a set of oriented Haar features to extract low-level motion and appearance information from images. We proposed a multidimensional boosting regression algorithm which can handle efficiently the high dimensionality of the output space. We tested our approach on calibrated video and motion capture walking sequences, and showed how it outperforms alternative approaches to multidimensional boosting regression.

7. Acknowledgements

Part of this work was conducted while the first author was an intern at Honda Research Institute in 2005. Work at UCLA was sponsored by ONR N00014-03-1-0850:P0001 and AFOSR FA9550-06-0138/E-16-V91-G2.

References

[1] A. Agarwal and B. Triggs. 3d human pose from silhouettes by relevance vector regression. *CVPR*, vol 2:pp. 882–888, 2004.

[2] A. Agarwal and B. Triggs. Monocular human motion capture with a mixture of regressors. *IEEE Workshop on Vision for Human-Computer Interaction*, 2005.

[3] A. Elgammal and C. S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. *CVPR*, 2004.

[4] R. Fablet and M. J. Black. Automatic detection and tracking of human motion with a view-based representation. *ECCV*, 2002.

[5] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient matching of pictorial structures. *CVPR*, vol 2:pp. 2066–2074, 2000.

[6] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.

[7] K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3d structure with a statistical image-based shape model. *ICCV*, 2003.

[8] G. Hua, M.-H. Yang, and Y. Wu. Learning to estimate human pose with data driven belief propagation. *CVPR*, vol 2:pp. 747–754, 2005.

[9] S. Ioffe and D. A. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision*, vol 43, no. 1:pp. 45–68, 2001.

[10] L.Brieman, J.H.Friedman, R.A.Olshen, and C.J.Stone. *Classification and Regression Trees*. Wadsworth&Brooks, 1984.

[11] M. W. Lee and I. Cohen. Proposal driven MCMC for estimating human body pose in static images. *CVPR*, 2004.

[12] G. Mori, X. Ren, A. A. Efros, and J. Malik. Recovering human body configurations: Combining segmentation and recognition. *CVPR*, vol 2:pp. 326–333, 2004.

[13] D. Ramanan, D. A. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. *CVPR*, 2005.

[14] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. *ECCV*, vol 4:pp. 700–714, 2002.

[15] R. Rosales and S. Sclaroff. Learning body pose via specialized maps. *In NIPS*, 2002.

[16] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. *ICCV*, vol 2:pp. 750–757, 2003.

[17] L. Sigal and M. J. Black. Humaneva: Syncronized video and motion capture dataset for evaluation of articulated human motion. *Brown Univertsity TR*, 2006.

[18] L. Sigal, M. Isard, B. H. Sigelman, and M. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. *In NIPS*, pages 1539–1546, 2003.

[19] C. Smichisescu. Learning to reconstruct 3d human motion from bayesian mixture of experts. *CVPR*, 2005.

[20] A. Thayananthan, R. Navaratnam, B. Stenger, P. H. S. Torr, and R. Cipolla. Multivariate relevance vector machines for tracking. *ECCV*, 2006.

[21] A. Torralba, K. P. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. *CVPR*, 2004.

[22] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *ICCV*, pages pp. 734–741, 2003.

[23] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:780–785, 1997.

[24] Y. Yacoob and M. J. Black. Parameterized modeling and recognition of activities. *Computer Vision and Image Understanding*, 1999.

[25] S. K. Zhou, B. Georgescu, X. S. Zhou, and D. Comaniciu. Image based regression using boosting method. *ICCV*, 2005.