Computationally Efficient Regression on a Dependency Graph for Human Pose Estimation

Kota Hara and Rama Chellappa
Center for Automation Research, University of Maryland, College Park, MD 20742
{kotahara,rama}@umiacs.umd.edu

Abstract

We present a hierarchical method for human pose estimation from a single still image. In our approach, a dependency graph representing relationships between reference points such as body joints is constructed and the positions of these reference points are sequentially estimated by a successive application of multidimensional output regressions along the dependency paths, starting from the root node. Each regressor takes image features computed from an image patch centered on the current node's position estimated by the previous regressor and is specialized for estimating its child nodes' positions. The use of the dependency graph allows us to decompose a complex pose estimation problem into a set of local pose estimation problems that are less complex. We design a dependency graph for two commonly used human pose estimation datasets, the Buffy Stickmen dataset and the ETHZ PASCAL Stickmen dataset, and demonstrate that our method achieves comparable accuracy to state-of-the-art results on both datasets with significantly lower computation time than existing methods. Furthermore, we propose an importance weighted boosted regression trees method for transductive learning settings and demonstrate the resulting improved performance for pose estimation tasks.

1. Introduction

Human pose estimation has been a widely studied topic in the computer vision community. Most of the early methods work on silhouettes extracted by background subtraction to reduce the complexity of the problem. However, reliably extracting silhouettes is itself a difficult task in practical settings and requires background images. Recently, the focus of the community has shifted toward pose estimation from a single still image in cluttered backgrounds. Although some of the techniques from the silhouette-based algorithms can be applied, the task is significantly more difficult, generating new challenges to address.

Most of the existing methods for pose estimation from a single image, including many state-of-the-art methods, are based on a pictorial structure model, which was first proposed in [1] for general computer vision problems and later applied to the pose estimation problem in [9]. The pictorial structure model represents a human body by a combination of body parts with spring-like constrains between those parts to enforce kinematically plausible spatial configurations. The inference is done by first evaluating the likelihood of each body part's locations on the image and then finding the most plausible configuration. If the model forms a tree structure, the globally optimum solution is efficiently found by dynamic programming.

Despite their successes, pictorial structure models have some problems. First, detecting body parts such as limbs, torso and head is challenging in a real-world scenario due to noisy backgrounds, occlusion and variation in appearances and poses. Most of the efforts have been devoted to building reliable body part detectors; however, they tend to be finely tuned to a specific dataset. Second, it is apparent that a simple pictorial structure model does not produce sufficiently good results and thus many efforts have concentrated on extending the basic pictorial structure model to more complex ones, requiring extensive computations.

In this paper, we propose a novel solution for the human pose estimation problem, which we call Regression on a Dependency Graph (RoDG). RoDG does not rely on detectors for each body part nor requires computationally expensive optimization methods. In RoDG, a dependency graph representing relationships between reference points such as body joints is specified and the positions of these reference points are sequentially estimated by a successive application of multidimensional output regression along the dependency paths, starting from the root node. Each regressor takes image features computed from an image patch centered on the current node's position estimated by the previous regressor and is specialized for estimating its child nodes' positions. The use of the dependency graph allows us to decompose a complex pose estimation problem into a set of local pose estimation problems that are much less

complex. In the training phase, those regressors are independently trained using images of people with ground-truth joint locations.

Most regression methods for the human pose estimation task [4, 2, 24] learn a single regressor mapping an image patch containing an entire human body region to all of the pose parameters. A drawback of this approach is that image patches have to be large enough to cover all possible poses and thus are dominated by a lot of background regions, making regression problems complex. In contrast, the size of the image patches in our approach is designed to contain mostly foreground regions that are sufficient to estimate local poses, reducing the complexity of the mapping problems.

RoDG is simple, versatile and significantly faster than existing approaches, yet achieves accuracy comparable to state-of-the-art on two popular benchmarks, the Buffy Stickmen dataset[1] and the ETHZ PASCAL Stickmen dataset[2]. We also propose an importance weighted variant of boosted regression trees for transductive learning settings and demonstrate its effectiveness for the human pose estimation task.

2. Related work

Many existing approaches to human pose estimation from a still image are based on a pictorial structure model. The focus of current research has been in 1) extending the models to a non-tree structures with efficient inference procedures and 2) improving body part detectors. Ren et al.[16] introduced pair-wise constraints between parts and use Integer Quadratic Programming to find the most probable configuration, however, their part detectors relied on simple line features. Andriluka et al.[3] used discriminatively trained part detectors to detect parts from images with complex backgrounds.

Instead of relying on a single model, Sapp et al.[18] proposed a coarse-to-fine cascade of pictorial structure models. In this approach, the coarser models are trained to efficiently prune implausible poses as much as possible while preserving the true poses for the finer level of pictorial structure models that are more accurate but computationally expensive. Sun et al.[19] extended the tree models of [18] to loopy models and presented an efficient and exact inference algorithm based on branch-and-bound.

Yang and Ramanan [26] proposed a mixture of templates for each part. They introduced a score term for representing co-occurrence relations between the mixtures of parts in a scoring function of the pictorial structure model and achieved impressive results. Ukita [21] extended [26] by introducing contour-based features to evaluate parts connectivities and achieved state-of-the-art results with at most four times the computation time of [26].

Several approaches to human pose estimation from cluttered images that do not use pictorial structure models [22, 11, 14, 2] have been developed. [22] applied MCMC technique to find the MAP estimate of the 3-dimensional pose. [11, 14] extended the Implicit Shape Model of [13] to the human pose estimation task by allowing voting in a pose parameter space.

Transductive learning was first applied to human pose estimation in [24] where the authors proposed importance weighted variants of kernel regression and twin Gaussian process model to remove the biases in the training set.

3. Method - Regression on a Dependency Graph

Let us denote I for an image, $p_i = (x, y)$ for a pixel location of the i-th key point in the image, where $i \in \{1, \ldots, K\}$. The key points may correspond to anatomically defined points of a human body or arbitrary defined reference points. A dependency graph on the key points is manually designed based on the anatomical structure of the human body. For notational simplicity, we assume p_1 corresponds to the root node. Each adjacent pair of nodes (i, j) in the graph has the following dependency:

$$p_j = s \cdot f_{i,j}(p_i, I, s) + p_i \tag{1}$$

where i and j are a parent and child node, respectively, s is the scale parameter and $f_{i,j}$ is a function that outputs a vector. Given a root node position p_1, scale s and an image I, we can determine subsequent $\{p_2, \ldots, p_K\}$ by successively applying Eq.(1) along all the graph paths.

Each function $f_{i,j}$ is defined as follows:

$$f_{i,j}(p_i, I, s) = g_{i,j}(h(p_i, I, s)) \tag{2}$$

where $g_{i,j}$ is a regressor and $h(p_i, I, s)$ is a predefined function which computes the image features from an image patch centered on p_i at scale s. The size of the image patches is designed to be sufficiently large to contain all possible p_j, however, it should not be larger than necessary.

Each regressor $g_{i,j}$ is independently trained from a set of images with ground-truth annotations of $\{p_1, \ldots, p_K\}$ and s. Input features for each regressor are computed by the same h. A target vector for each regressor is the relative location of p_j with respect to p_i normalized by s and can be computed by solving Eqs.(1) and (2) for $g_{i,j}$:

$$g_{i,j}(h(p_i, I, s)) = (p_j - p_i)/s \tag{3}$$

Note that each regressor $g_{i,j}$ is a multidimensional output regressor as the output is a 2-dimensional vector. Furthermore, for a parent node i that has more than two child

[1] http://www.robots.ox.ac.uk/~vgg/data/stickmen/

[2] http://groups.inf.ed.ac.uk/calvin/ethz_pascal_stickmen/

nodes $\{j_1, \ldots, j_L\}$, we define a single multidimensional output regressor that computes an output for each child node at once from the same input:

$$g_i(\cdot) = (g_{i,j_1}(\cdot), \ldots, g_{i,j_L}(\cdot)) \in \mathbb{R}^{2L} \quad (4)$$

In Fig.1 left, we show an instance of the dependency graph designed for the datasets used in the experiments. The non-root nodes of the graph correspond to a set of body joints used to represent a human body pose in the dataset. In Fig.1 right, the red box represents a detection window given by an upper body detector. The root node corresponds to the center of the detection window while the other nodes correspond to endpoints of sticks representing a head, torso, upper and lower arms. The scale s is determined by the ratio between the size of the detection window and a predefined canonical window size.

The dependency graph is designed by taking into account the anatomical structure of the human body and also the pose representation adopted by the target datasets. For instance, we make both nodes 7 and 8 depend on node 6 in the graph as they represent body points that are close to each other and thus are contained by the image patch centered on p_6. Similarly, we make nodes 2,3,4,5,6,10 depend on node 1 as their positions do not vary significantly with respect to p_1. Designing an optimum dependency graph for a given task is an interesting topic which will be considered in future.

The details of the training and testing steps on this structure are presented in Section 5. Note that RoDG is quite general and applicable to other tasks such as facial points localization and hand pose estimation by properly designing the dependency graphs.

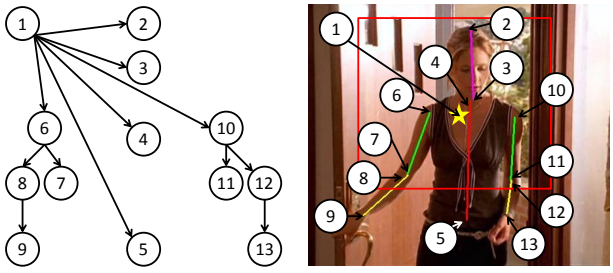


Figure 1. Left: Dependency graph, Right: Semantics of the nodes. The red box is a detection window and the yellow star is the center of the detection window.

4. Multidimensional Output Regression on Weighted Training Samples

Multidimensional output regression allows us to train a single model that outputs target vectors instead of independently training a single model for each output dimension. We denote a set of training samples by $\{\mathbf{t}_i, \mathbf{x}_i\}_{i=1}^N$, where $\mathbf{t}$ is a target vector and $\mathbf{x}$ is an input vector. Furthermore, we denote the weight of the i-th training sample as w_i. All the weights are set to 1 except for in the transductive learning setting (Section 4.3).

The goal of regression is to learn a function $F^*(\mathbf{x})$ such that the expected value of a certain loss function $\Psi(\mathbf{t}, F(\mathbf{x}))$ is minimized:

$$F^*(\mathbf{x}) = \underset{F(\mathbf{x})}{\operatorname{argmin}} \, \mathrm{E}[\Psi(\mathbf{t}, F(\mathbf{x})] \quad (5)$$

By approximating the above expected loss by empirical loss, we obtain

$$F^*(\mathbf{x}) = \underset{F(\mathbf{x})}{\operatorname{argmin}} \sum_{i=1}^{N} w_i \Psi(\mathbf{t}_i, F(\mathbf{x}_i)). \quad (6)$$

4.1. Multidimensional Output Regression Tree on Weighted Training Samples

We propose a multidimensional output regression tree on weighted training samples and use it as a building block for the gradient boosting procedure which is presented in Section 4.2. The multidimensional output regression tree is a non-linear regression model represented as follows:

$$H(\mathbf{x}; \mathcal{A}, \mathcal{R}) = \sum_{k=1}^{K} \mathbf{a}_k \mathbb{1}(\mathbf{x} \in r_k) \quad (7)$$

where $\mathbb{1}$ is an indicator function, $\mathcal{R} = \{r_1, \ldots, r_K\}$ is a set of disjoint partitions of the input space and $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_K\}$ is a set of vectors. Each $\mathbf{a}_k$ is computed as the *weighted* mean of the target vectors of the training samples that fall into r_k.

In the training phase, the regression tree is grown by recursively partitioning the input space, starting from a root node which corresponds to the entire input space. Subsequent partitions are applied to one of the leaves. Throughout the growth of the tree, $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{K'}\}$, where K' is the number of leaves at the time and the weighted sum of squared error for each leaf node k is computed as follows:

$$S_k = \sum_{i \in r_k} w_i ||\mathbf{t}_i - \mathbf{a}_k||_2^2 \quad (8)$$

Then the weighted sum of squared error on the entire training data is given by $S = \sum_{k=1}^{K'} S_k$.

At each partitioning stage, the leaf with the largest weighted sum of squared error is selected for partitioning. A binary split rule defined by an index of the input dimension and a threshold is selected among all possible split rules such that the reduction in S is maximized. When computing the weighted means and sum of squared errors, an efficient incremental algorithm such as [23] is used. The recursive partitioning stops when K leaves are generated, where K is a predefined parameter.

4.2. Multidimensional Output Boosted Regression Trees on Weighted Training Samples

A gradient boosting machine [10] is an algorithm to construct a strong regressor from an ensemble of weak regressor. In this paper, we use the proposed weighted variant of multidimensional output regression tree as a weak regressor. The strong regressor $F(\mathbf{x})$ is expressed as an ensemble of regression trees H:

$$F(\mathbf{x}; P) = \sum_{m=0}^{M} H(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m) \tag{9}$$

where $P = \{\mathcal{A}_m, \mathcal{R}_m\}_{m=0}^{M}$ represents the set of regression trees' parameters.

In the training phase, the gradient boosting algorithm tries to minimize the function in Eq.(6) by sequentially adding a new regression tree H at each stage m, where $m = 0$ to M. At each stage except for $m = 0$, a set of the parameters of the tree is determined such that the updated model maximally reduces the loss:

$$(\mathcal{A}_m, \mathcal{R}_m) = \underset{\mathcal{A},\mathcal{R}}{\operatorname{argmin}} \sum_{i=1}^{N} w_i \Psi(\mathbf{t}_i, F_{m-1}(\mathbf{x}_i) + H(\mathbf{x}_i; \mathcal{A}, \mathcal{R})) \tag{10}$$

Then the learned regression tree is added to the current model,

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + H(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m). \tag{11}$$

For $m = 0$, $F_0(\mathbf{x})$ is the *weighted* mean target vector of all training samples.

Choosing the squared error loss function $\Psi(\mathbf{t}, F(x)) = ||\mathbf{t} - F(x)||_2^2$ and the weighted regression trees as the weak regressor, we obtain Algorithm 1, where ν is a shrinkage parameter to prevent overfitting. Each tree H is trained using residual $\tilde{\mathbf{t}}$ of each training sample recomputed at each iteration as target vectors. A non-weighted version of the algorithm is also described in [4].

Algorithm 1 Multidimensional Output Boosted Regression Trees on Weighted Training Samples

1: $F_0(\mathbf{x}) = \bar{\mathbf{t}}$ ▷ weighted mean
2: **for** $m = 1$ to M **do**
3: $\quad \tilde{\mathbf{t}}_i = \mathbf{t}_i - F_{m-1}(\mathbf{x}_i), i = 1, \ldots, N$
4: $\quad (\mathcal{A}_m, \mathcal{R}_m) = \underset{\mathcal{A},\mathcal{R}}{\operatorname{argmin}} \sum_{i=1}^{N} w_i ||\tilde{\mathbf{t}}_i - H(\mathbf{x}_i; \mathcal{A}, \mathcal{R})||_2^2$
5: $\quad F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu H(\mathbf{x}; \mathcal{A}_m, \mathcal{R}_m)$
6: **end for**

4.3. Importance Weighted Boosted Regression Trees

In a transductive learning setting, (unlabeled) testing samples are available during the training phase along with labeled training samples. When the test samples and training samples are drawn from different probability distributions, the regressor trained solely on the training samples is not optimal for the given test samples. One of the possible remedies to this problem is realized by weighting each training sample by an *importance weight* w such that the new distribution formed by the weighted training samples resembles the distribution of testing samples. This is accomplished by setting the importance weight of the i-th training sample as $w_i = p_{te}(\mathbf{x}_i)/p_{tr}(\mathbf{x}_i)$, where p_{te} and p_{tr} are probability density functions of the testing samples and training samples respectively. The proposed weighted variant of the boosted regression trees can work with any method that estimate importance weights. In this paper we adopt RuLSIF [25] owing to its impressive performance.

Instead of working on the entire testing samples at once, we first cluster the testing samples into several clusters by the k-means algorithm and for each cluster we independently estimate the importance weights and train a regressor. This would make the probability density of each cluster simpler and ease the estimation of the importance weights. Furthermore, we transform the testing samples to N_{tr} dimensional vectors by computing a kernel matrix $K = (k(\mathbf{x}_i^{te}, \mathbf{x}_j^{tr}))_{i,j}, i = 1, \ldots, N_{te}, j = 1, \ldots, N_{tr}$ where N_{te} and N_{tr} are the number of the testing and training samples respectively. This feature transformation and clustering was found to improve the accuracy.

5. Experiments

We tested our algorithm on publicly available datasets for the upper body pose estimation task. The performance is measured by the Percentage of Correctly estimated body Parts (PCP). A comparison with existing works reveals the advantages of our method.

5.1. Datasets

We use the Buffy Stickmen dataset and the ETHZ PASCAL Stickmen dataset to evaluate our method. Both datasets have the same representation of poses and provide the same protocol to measure the performance. A body pose is represented by 6 sticks representing a torso, head, upper arms and lower arms (see Fig. 1). Each stick is represented by the locations of the two endpoints. Both datasets come with detection windows containing upper bodies obtained by an upper body detector. The performance is measured only on the images with detection windows, allowing the separation of the human detection task from the pose estimation task. As two endpoints of each stick are annotated without consistent ordering, we manually swap two endpoints if necessary.

The Buffy Stickmen dataset has 748 images taken from the TV show *Buffy the Vampire Slayer* and it is very challenging due to highly cluttered backgrounds. However, the

same subjects with same clothing occasionally appear in both training and testing set which makes the task easier. Among 748 images, 276 images are specified as testing data while 472 images are used for training. In the first release of the dataset, 85.1% of the images in the testing set come with detection windows while 95.3% come with detection windows obtained by an improved detector in the latest release.

The PASCAL Stickmen dataset contains images taken from the PASCAL VOC 2008 trainval release. Unlike the Buffy Stickmen dataset, it consists mainly of 549 amateur photographs with unconstrained illumination, severe occlusion and low image quality making this dataset more challenging than the Buffy dataset. In the first release, 65.6% of the images come with detection windows while 75.1% in the latest release with the improved detector. Note that the PASCAL dataset is used only for testing.

The performance of the pose estimation algorithms is measured using PCP. Each body part is represented as a stick and its estimate is considered correct if its endpoints lie within $100t\%$ of the length of the ground-truth stick from their ground-truth locations. We denote PCP with $t = 0.5$ by $\mathrm{PCP}_{0.5}$.

Both datasets come with a tool to compute the PCP, however, it was recently pointed out in [15] that the tool does not exactly compute the above defined PCP, leading to erroneously higher PCP. As most of the existing works report PCP on the detection windows in the first releases of the dataset using this tool, we also report PCP using the same tool. To facilitate future comparison, we also report the correct PCP computed by a fixed version of the tool[3] on the updated detection windows provided in the latest releases. To eliminate any confusion we precisely define a condition that an estimated part (i.e. stick) has to satisfy to be considered as correctly localized:

$$\begin{aligned}&(||E_1 - G_1||_2 \leq t \cdot L \wedge ||E_2 - G_2||_2 \leq t \cdot L)\\ &\qquad\qquad \vee \\ &(||E_1 - G_2||_2 \leq t \cdot L \wedge ||E_2 - G_1||_2 \leq t \cdot L)\end{aligned} \tag{12}$$

where (E_1, E_2) and (G_1, G_2) are locations of the two endpoints of the estimated and ground-truth stick, respectively, and $L = ||G_1 - G_2||_2$.

5.2. Implementation Details

In order to obtain the ground-truth of the root node, a set of detection windows containing the annotated upper bodies in the training images are first obtained by running the same upper body detector used to obtain the detection windows for the testing set. Each image has exactly one annotated human. Detection windows are obtained for 345 out of 472 training images in the Buffy training set[4]. The scale s for each sample is determined by the width of the detection window divided by 64. The ground-truth for the other nodes are included in the dataset.

The image patches from which $h(p_i, I, s = 1)$ computes image features is set to 64×64 pixel rectangular region whose center is located at p_i. From each patch, we compute multiscale HOG [5] with cell size 8, 16, 32 and 2×2 cell blocks. The orientation histogram for each cell is computed with unsigned gradients with 9 orientation bins. The dimensionality of the resultant HOG feature is 2124. For an arbitrary s, the image patch size is scaled by s while keeping the center location unchanged.

In its original form the dependency graph (Fig.1) requires 5 regressor, namely, $g_{1,\{2,3,4,5,6,10\}}$, $g_{6,\{7,8\}}$, $g_{10,\{11,12\}}$, $g_{8,9}$ and $g_{12,13}$. In order to exploit the symmetric structure of the human body, we train a shared regressor for $g_{6,\{7,8\}}$ and $g_{10,\{11,12\}}$ by horizontally flipping the training samples for the key points on the right side of the body. In the testing time, the same regressor is used for both sides but for the right side both the input patch and output vector need to be horizontally flipped. We do the same for $g_{8,9}$ and $g_{12,13}$. This procedure practically doubles the number of the training samples. For $g_{1,\{2,3,4,5,6,10\}}$, we also double the number of the training samples by appropriately mirroring each training sample.

For boosted regression trees, the number of the leaves in the regression trees K is set to 5 and the shrinkage parameter ν is set to 0.1 following the suggestion in [12]. Through cross-validation on the training set, it is observed that the error keeps decreasing as the number of trees increases. Thus, we empirically set the number of trees M to 2000 for $g_{1,\{2,3,4,5,6,10\}}$ and 1000 for the rest. The regressors are trained on the Buffy training set and the same regressors are used for testing on both Buffy testing set and PASCAL dataset.

5.3. Results

As our RoDG works with any multidimensional output regression methods, we also test RoDG with Kernel Partial Least Squares (KPLS) [17], Partial Least Squares (PLS) [6], Lasso [7] and Multivariate RVM (MRVM) [20]. The parameters of those regression methods are determined by 5-fold cross validation.

In Table 1, we show the results on the Buffy dataset evaluated with the PCP tool provided in the dataset and the detection windows in the initial release of the dataset, while in Table 2, we show the results with the fixed PCP tool and the updated detection windows in the latest release.

As can be seen from Table 1, the RoDG-Boost achieves the second best total $\mathrm{PCP}_{0.5}$ next to [21] with significantly lower computation time (Table 5). Note that unlike some of the previous works, RoDG does not require external training data nor exploit color information. For reference, we

[3]The fixed tool will be available on the author's website.

[4]We thank Marcin Eichner for providing the results.

also compare our methods with [21][5] using a stricter criteria (total $\mathrm{PCP}_{0.2}$) and found out that RoDG-Boost outperforms [21] with a large margin (RoDG-Boost:63.0, [21]:58.2). This result indicates that the ranking of performance varies depending on the PCP threshold, thus comparisons should also be made by PCP-curves obtained by varying the PCP threshold. Table 2 shows that RoDG-Boost and RoDG-KPLS outperform the existing method with a large margin.

The PCP values on the first setting are higher than those on the second setting due to the flaw of the original PCP tool. The correct PCP scores reveal that there is still much room for improvement, especially for lower arms. In Fig.2(a), we plot the PCP curves on the Buffy testing set with the second setting. RoDG-Boost consistently outperforms RoDG-KPLS when PCP threshold is less than 0.47 and both methods significantly outperform the state-of-the-art. We encourage future comparisons on this new setting with PCP curves.

Table 1. $\mathrm{PCP}_{0.5}$ on Buffy with the original PCP tool and detection windows

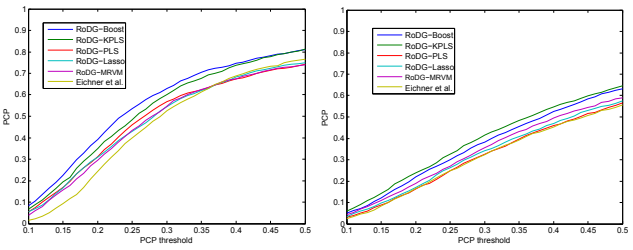
	total	torso	u.arms	l.arms	head
RoDG-Boost	89.8	99.6	96.8	73.0	99.6
RoDG-KPLS	88.2	100	96.6	68.7	98.7
RoDG-MRVM	87.5	99.6	97.2	67.0	97.0
RoDG-LASSO	86.7	100	96.7	63.6	99.6
RoDG-PLS	87.2	100	97.5	65.3	97.9
Ukita [21]	90.3	100	97.5	73.9	98.9
Yang [26]	89.1	100	96.6	70.9	99.6
Zuffi [27]	85.6	99.6	94.7	62.8	99.2
Sun [19]	85.7	99.6	93.8	63.9	99.2
Sapp [18]	85.5	100	95.3	63.0	96.2
Andriluka [3]	83.1	97.5	92.7	59.6	95.7

Table 2. $\mathrm{PCP}_{0.5}$ on Buffy with the updated PCP tool and detection windows

	total	torso	u.arms	l.arms	head
RoDG-Boost	81.1	98.5	92.8	51.5	99.2
RoDG-KPLS	81.2	99.2	92.8	51.3	99.6
RoDG-MRVM	76.9	98.9	91.8	40.5	97.7
RoDG-LASSO	74.6	98.5	89.7	35.4	98.9
RoDG-PLS	74.2	99.6	90.5	33.5	97.7
Eichner [8]	76.7	99.6	81.9	50.0	96.6

In Tables 3 and 4, we show the results on the PASCAL dataset with the two settings. We achieve state-of-the-art results on both settings. The PCPs on the PASCAL are much lower than that on Buffy. We argue that the reasons are 1) the PASCAL dataset is more difficult due to more complex poses, more challenging occlusions and blur, 2) the similarity between the testing and training sets in the Buffy dataset favors PCP on the Buffy dataset. In Fig.2(b), we plot the PCP curves on the PASCAL dataset with the second setting. RoDG-KPLS consistently outperforms RoDG-Boost, however, RoDG-KPLS is much more computationally expensive due to KPLS execution (Table 5).

[5]We thank Norimichi Ukita for providing the results.



(a) Buffy stickmen (b) PASCAL Stickmen

Figure 2. PCP curves with the second setting (best viewed in color)

Table 3. $\mathrm{PCP}_{0.5}$ on PASCAL with the original PCP tool and detection windows

	total	torso	u.arms	l.arms	head
RoDG-Boost	79.2	100	87.8	50.4	98.9
RoDG-KPLS	78.8	99.7	87.0	50.4	98.6
RoDG-MRVM	77.5	99.7	86.0	47.5	98.1
RoDG-LASSO	76.4	100	86.7	44.4	96.1
RoDG-PLS	76.3	99.7	87.0	43.8	96.9
Sun [19]	78.8	99.7	81.4	55.4	99.4
Sapp [18]	77.2	100	87.1	49.4	90.0
Andriluka [3]	71.8	96.4	77.8	47.0	85.0

Table 4. $\mathrm{PCP}_{0.5}$ on PASCAL with the updated PCP tool and detection windows

	total	torso	u.arms	l.arms	head
RoDG-Boost	63.3	91.5	75.1	27.8	82.3
RoDG-KPLS	64.5	87.1	77.2	30.5	84.7
RoDG-MRVM	59.6	87.1	71.5	26.1	75.5
RoDG-LASSO	57.4	89.6	69.4	22.1	71.6
RoDG-PLS	56.5	88.8	72.1	18.0	69.9
Eichner [8]	55.7	96.6	60.6	27.3	61.9

Table 5 presents approximate computation times of each method to process one image. Note that the computation time of previous methods are taken from their original papers or websites and thus are not obtained by running on the same computer, however, they give a rough idea on compu-

tational requirements of each method. All RoDGs are run on Xeon 3.6GHz CPU machine. All RoDGs run significantly faster than all the previous methods.

Table 5. Computation time per image. Left: our methods, Right: existing methods

method	time	method	time
RoDG-Boost	23 msec.	Ukita [21]	4 sec.
RoDG-KPLS	193 msec.	Yang [26]	1 sec.
RoDG-PLS	13 msec.	Zuffi [27]	a few min.
RoDG-LASSO	13 msec.	Sun [19]	300 sec.
RoDG-MRVM	15 msec.	Sapp [18]	300 sec.
		Andriluka [3]	50 sec.
		Eichner [8]	6.6 sec.

Representative results of RoDG-Boost on Buffy and PASCAL are shown in Fig.3 and Fig.4, respectively.

Transductive learning results

We evaluate the performance of RoDG with our importance weighted boosted regression trees in transductive settings. As the fixed PCP tool is more adequate to compare the performance of the methods, we conduct experiments only using the second setting. For RuLSIF, we use the same parameter settings employed in [25]. We use a Gaussian kernel with $\sigma = 10$ for feature transformation and set the number of clusters to 10 and 20 for Buffy and PASCAL, respectively. The parameters of the gradient boosting are kept the same.

Tables 6 and 7 show the results on the Buffy and PASCAL dataset, respectively. The first row presents the results of non-transductive settings, the second row, the results of transductive settings without clustering and the third row presents the results with clustering. On the Buffy dataset, the PCP clearly improves while on the PASCAL dataset, RuLSIF degrades the performance but RuLSIF-cluster recovers the loss.

Table 6. $\mathrm{PCP}_{0.5}$ of importance weighted boosted regression trees on Buffy

	total	torso	u.arms	l.arms	head
Base	81.1	98.5	92.8	51.5	99.2
RuLSIF	81.6	98.9	92.6	53.2	99.2
RuLSIF-clstrs	82.5	98.9	93.5	54.9	99.2

Table 7. $\mathrm{PCP}_{0.5}$ of importance weighted boosted regression trees on PASCAL

	total	torso	u.arms	l.arms	head
Base	63.3	91.5	75.1	27.8	82.3
RuLSIF	63.0	90.3	75.2	28.8	79.9
RuLSIF-clstrs	63.4	90.3	75.5	27.9	83.0

6. Conclusion

In this paper, we presented an algorithm for human pose estimation from a still image based on successive application of multidimensional output regressions on a dependency graph. The pose estimation problem was divided into a set of local pose estimation problems and solved sequentially from the root node of the graph. The method is a competitive alternative to pictorial structure-based methods for human pose estimation. On the two popular benchmarks, Buffy Stickmen and ETHZ PASCAL Stickmen, our method achieves comparable accuracy to state-of-the-art result with significantly lower computation time. Furthermore, we proposed boosted regression trees for importance weighted samples and applied it to transductive learning settings for human pose estimation.

Acknowledgements. This research was supported by a MURI grant from the US Office of Naval Research under N00014-10-1-0934.

References

[1] M. A. Fischler and R. A. Elschlager. The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, 1973.

[2] A. Agarwal and B. Triggs. A Local Basis Representation for Estimating Human Pose from Cluttered Images. *ACCV*, 2006.

[3] M. Andriluka, S. Roth, and B. Schiele. Discriminative Appearance Models for Pictorial Structures. *IJCV*, 2011.

[4] A. Bissacco, M.-H. Yang, and S. Soatto. Fast Human Pose Estimation using Appearance and Motion via Multidimensional Boosting Regression. *CVPR*, 2007.

[5] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. *CVPR*, 2005.

[6] S. de Jong. Simpls: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 1993.

[7] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *The Annals of Statistics*, 2004.

[8] M. Eichner, M. Marin-Jimenez, A. Zisserman, and V. Ferrari. 2D Articulated Human Pose Estimation and Retrieval in (Almost) Unconstrained Still Images. *IJCV*, 2012.

[9] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Matching of Pictorial Structures. *CVPR*, 2000.

[10] J. H. Friedman. Greedy Function Approximation: a Gradient Boosting Machine. *Annals of Statistics*, 2001.

[11] K. Hara and T. Kurokawa. Human Pose Estimation Using Patch-based Candidate Generation and Model-based Verification. *Face and Gesture (FG)*, 2011.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Second Edition*. Springer, 2009.

[13] B. Leibe, A. Leonardis, and B. Schiele. Robust Object Detection with Interleaved Categorization and Segmentation. *IJCV*, 2008.

[14] J. Müller and M. Arens. Human Pose Estimation with Implicit Shape Models. *ARTEMIS*, 2010.

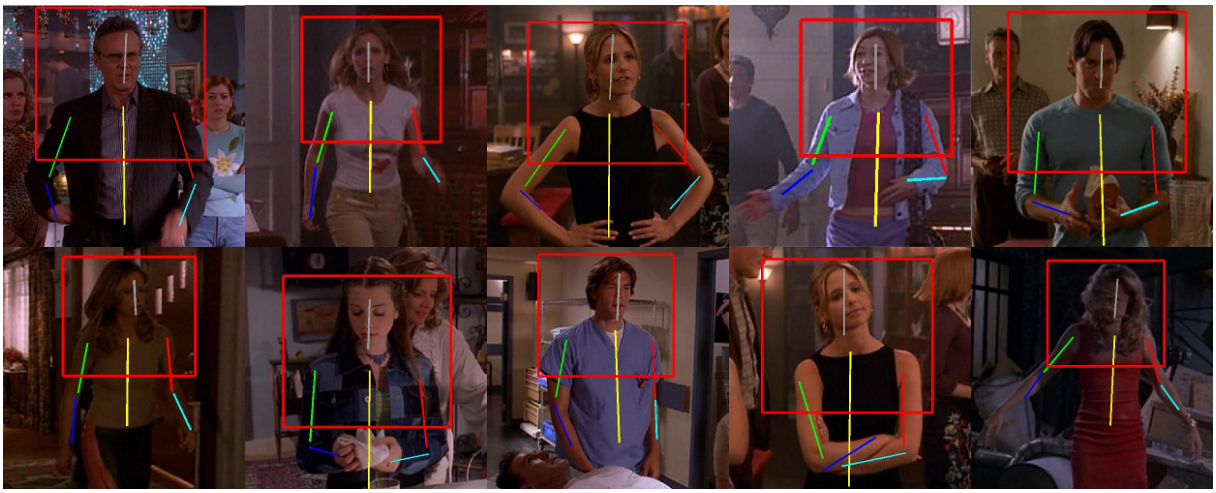


Figure 3. Representative results of RoDG-Boost on Buffy Stickmen dataset.

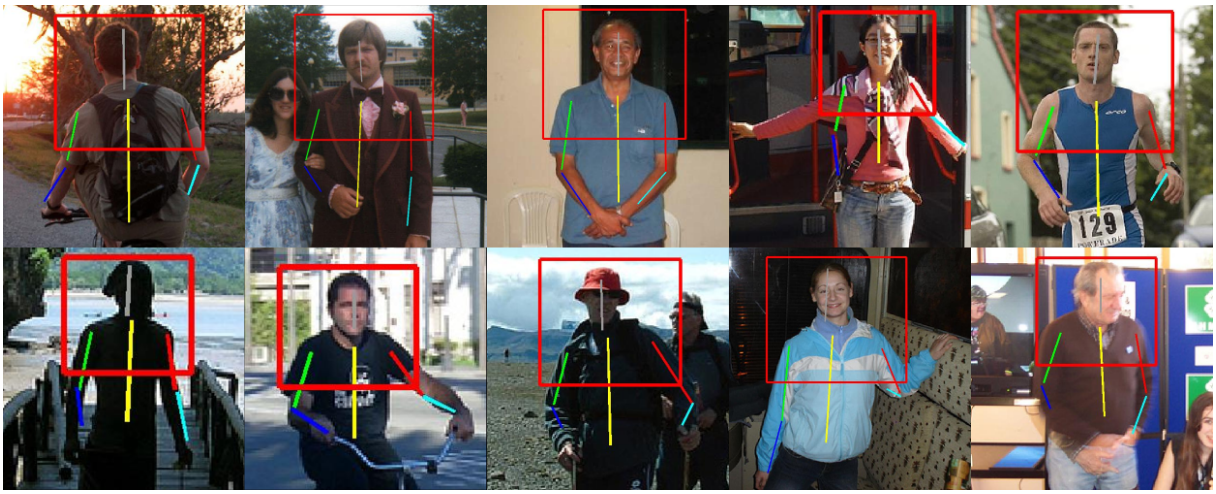


Figure 4. Representative results of RoDG-Boost on PASCAL Stickmen dataset. The last two columns show failure cases.

[15] L. Pishchulin, A. Jain, M. Andriluka, T. Thormählen, and B. Schiele. Articulated People Detection and Pose Estimation: Reshaping the Future. *CVPR*, 2012.

[16] X. Ren, A. C. Berg, and J. Malik. Recovering Human Body Configurations Using Pairwise Constraints Between Parts. *ICCV*, 2005.

[17] R. Rosipal and L. J. Trejo. Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space. *JMLR*, 2001.

[18] B. Sapp, A. Toshev, and B. Taskar. Cascaded Models for Articulated Pose Estimation. *ECCV*, 2010.

[19] M. Sun, M. Telaprolu, H. Lee, and S. Savarese. An Efficient Branch-and-bound Algorithm for Optimal Human Pose Estimation. *CVPR*, 2012.

[20] A. Thayananthan, R. Navaratnam, B. Stenger, P. H. Torr, and R. Cipolla. Multivariate Relevance Vector Machines for Tracking. *ECCV*, 2006.

[21] N. Ukita. Articulated Pose Estimation with Parts Connectivity Using Discriminative Local Oriented Contours. *CVPR*, 2012.

[22] M. W. Lee and I. Cohen. Proposal Maps Driven MCMC for Estimating Human Body Pose in Static Images. *CVPR*, 2004.

[23] D. H. D. West. Updating Mean and Variance Estimates: An Improved Method. *Communications of the ACM*, 1979.

[24] M. Yamada, L. Sigal, and M. Raptis. No Bias Left Behind: Covariate Shift Adaptation for Discriminative 3D Pose Estimation. *ECCV*, 2012.

[25] M. Yamada, T. Suzuki, T. Kanamori, H. Hachiya, and M. Sugiyama. Relative Density-ratio Estimation for Robust Distribution Comparison. *NIPS*, 2011.

[26] Y. Yang and D. Ramanan. Articulated Pose Estimation with Flexible Mixtures-of-Parts. *CVPR*, 2011.

[27] S. Zuffi, O. Freifeld, and M. J. Black. From Pictorial Structures to Deformable Structures. *CVPR*, 2012.