Articulated Pose Estimation with Parts Connectivity using Discriminative Local Oriented Contours

Norimichi Ukita
Nara Institute of Science and Technology
ukita@ieee.org

Abstract

This paper proposes contour-based features for articulated pose estimation. Most of recent methods are designed using tree-structured models with appearance evaluation only within the region of each part. While these models allow us to speed up global optimization in localizing the whole parts, useful appearance cues between neighboring parts are missing. Our work focuses on how to evaluate parts connectivity using contour cues. Unlike previous works, we locally evaluate parts connectivity only along the orientation between neighboring parts within where they overlap. This adaptive localization of the features is required for suppressing bad effects due to nuisance edges such as those of background clutter and clothing textures, as well as for reducing computational cost. Discriminative training of the contour features improves estimation accuracy more. Experimental results verify the effectiveness of our contour-based features.

1. Introduction

Articulated pose estimation in single images is presented in this paper. A number of recent methods employ pictorial structure models[8]. The models allow us to efficiently acquire a globally optimized geometric configuration of the whole parts of a target object (e.g. human body) in an image. The optimization is achieved so that the summation of the following scores is maximized:

Local appearance similarity of a part: The local appearance score of each part grows as its appearance cues are more similar to those at each location in an image.

Relative consistency between parts: The more probable the geometric configuration of a pair of parts, the relative consistency score gets higher.

While the pictorial structures are successful, they impose two severe limitations on their structures and score functions for efficient score computation. **1)** The structure of the model must be a tree with no loops, where relative consistency only between neighboring parts is evaluated. The tree structure allows dynamic programming to efficiently and globally optimize the locations of all parts. **2)** The pairwise score between the neighboring parts must be a quadratic function. In previous works, this limitation is satisfied so that only the cost of relative geometric deformation is evaluated with no appearance cues. With this limitation, distance transform can efficiently compute the max score among all possible locations of the neighboring parts.

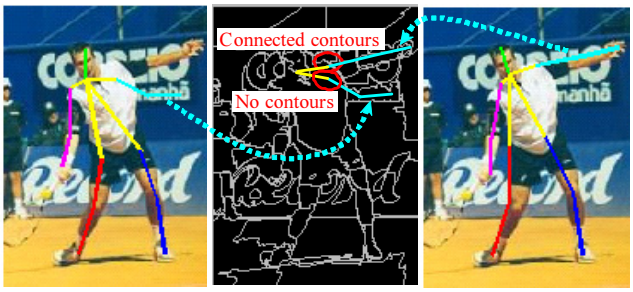


Figure 1. (Left) Typical estimation error with no appearance cues between neighboring parts. The left arm was located erroneously in a background, even though its contour was observed and connected to that of the torso (an upper red circle in the Middle). (Right) Effect of discriminatively trained local contour features between neighboring parts, which is a contribution of this paper.

Those limitations in the pictorial structure models cause the following problems. One of common problems is difficulty in good appearance cues for an individual part because the similarity score function must be generalized for a huge variety of the part appearance. The other problem is weak localization of neighboring parts only by the limitation in their deformable range. For example, human body parts are widely deformable (e.g. upper arms spin 360 degrees while they are connected to a torso).

To solve those problems, this paper proposes a scheme for evaluating appearance cues not only within each part but also between neighboring parts. We focus on how to evaluate parts connectivity using *edge contours* (Fig. 1).

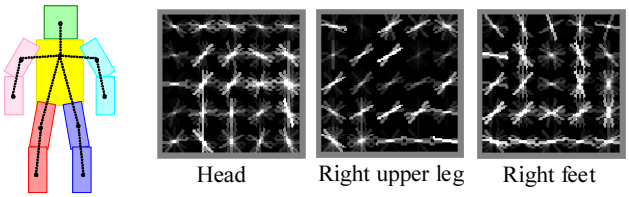


Figure 2. (Left) Tree-based articulated model for human pose estimation. Rectangles and dotted line segments depict nodes of their respective parts and links between them. One of the nodes is regarded as a root node; mainly head or torso nodes in a human body model. Each node has the pose parameters of the respective part (e.g. x-y locations, orientation θ, and size s). (Right) Examples of HOG features[5], each of which is used as the appearance template of the respective part. The HOG features are evaluated within the rectangles.

Our contribution is twofold: 1) contour features are trained discriminatively, and 2) they are evaluated locally just between neighboring parts for suppressing bad effects by nuisance edges and the rise in computational cost.

Our effective use of contour cues improves pose estimation accuracy, while the computational cost of the estimation process increases at most 4 times of the base model[31]. With a people image dataset[19], our method outperforms the base model[31] using the pictorial structure models in the correctly localized percentages of all body parts.

2. Related Work

Two main research issues exist in pose estimation, namely an *articulated model representation* and a *image feature representation*.

Articulated models are represented by graphical models, in general. In the graphical models, each node and link corresponds to a part and a physical connection between parts as illustrated in Fig. 2 (Left). Among all possible models, tree-based models[21], including pictorial structure models[8], are widely used because of their ability to represent a variety of shape structures and simplicity for obtaining optimal part locations.

Complex graphical models with loops[3, 30] can improve estimation accuracy and stability against background clutter. More links are more useful as proposed in [28, 13]. A known problem in those models is difficulty in acquiring optimal solutions because dynamic programming is inapplicable. Instead of by dynamic programming, optimal solutions in the complex models are acquired by other techniques; shortest path search[11], particle filtering and belief propagation[10, 26], integer quadratic programming[20], RANSAC[15], approximate search[28], and decomposition to simpler graphs[13].

Image features are fundamental issues not only in pose estimation but also in many computer vision and pattern recognition problems. For pose estimation of general objects, the image features should be generalized for representing a huge variety of part appearances, which are observed by different clothing and habitus in human pose estimation, for example. Such generalized features can be realized by binning/histogram-based representations (e.g. using shape contexts[17], histogram of oriented gradients (HOG)[7, 31]. Figure 2 (Right) shows the HOG templates of several human body parts. To maintain discriminativity as well as generality, the features of each part in training data are divided into several clusters and then trained individually (e.g. clustering based on 2D parts configurations[12, 31] and 3D parts configurations[4]). Using multimodal cues is also an effective approach (e.g. superpixels, color and size[18], edges and color-segmented regions[19], and contours, gradients, and color[22]). Recent advances have proven that discriminative training of part appearance can improve part distinguishability[21, 7, 1].

While many complex graphical models and multimodal image cues have been developed as described above, only a simple geometric deformation cue is given to the links of the models for evaluating parts connectivity. That is, few works have considered more complex but useful features for the links in the model. We focus on how to employ and optimize image features (i.e. contours, in this paper) for imposing useful constraints on the links so that neighboring parts are connected correctly.

Evaluating appearance cues between parts causes the rise in computational cost because distance transform cannot be employed in pictorial structure models. However, many approaches enable speeding up pose estimation; using lower-dimensional but discriminative descriptors[6], cascade[22]/hierarchical[32] models for coarse-to-fine search, and search space reduction based on person detectors[9, 7] and branch-and-bound pruning[24, 27].

3. Pictorial Structure Models

This section describes basic tree-based articulated part models. A tree is defined by a set of nodes, $\boldsymbol{V}$, and a set of links each of which connect two nodes, $\boldsymbol{E}$. In this paper, a human body model is used for human pose estimation. Figure 2 (Left) is its example. Each node has pose parameters that localize the respective part. By optimizing the pose parameters in accordance with a human pose in an image, pose estimation is achieved. The pose parameters are optimized by maximizing the score function below:

$$T(\boldsymbol{P}) = \sum_{i \in \boldsymbol{V}} S^i(\boldsymbol{p}_i) + \sum_{i,j \in \boldsymbol{E}} P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j), \quad (1)$$

where $\boldsymbol{p}_i$ and $\boldsymbol{P}$ denote a set of the pose parameters of i-th part and a set of $\boldsymbol{p}_i$ of all parts (i.e. $\boldsymbol{P} = \{\boldsymbol{p}_i | \forall i \in \boldsymbol{V}\}$).

$S^i(\boldsymbol{p}_i)$ and $P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$, which are prepared for each part i and pair of neighboring parts i and j, explained below.

A unary term $S^i(\boldsymbol{p}_i)$ is a similarity score of i-th part at $\boldsymbol{p}_i$. This term depends only on local appearance at $\boldsymbol{p}_i$. In our model, $S^i(\boldsymbol{p}_i)$ is the filter response using HOG features[5], each of which consists of 5×5 cells and 18 orientation bins:

$$S^i(\boldsymbol{p}_i) = F^i \cdot \phi(I, \boldsymbol{p}_i), \tag{2}$$

where F^i and $\phi(I, \boldsymbol{p}_i)$ denote the filter of i-th part and the HOG extracted from $\boldsymbol{p}_i$ in image I.

A pairwise term $P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$ is a spring-based score function between i-th and j-th parts, which has a greater value if the configuration $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ is highly probable.

In the tree-based model[8], the globally optimized pose parameters, $\hat{\boldsymbol{P}}$, can be acquired efficiently by dynamic programming in $O(l^2 n)$, where l is the discrete number of possible values in $\boldsymbol{p}_i$. Dynamic programming acquires the max score of (1) by passing the following message from every leaf node, i, to its parent node, j, recursively toward a root node:

$$M(i,j) = S^i(\boldsymbol{p}_i) + P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) + \sum_{c \in \boldsymbol{C}_i} M(c,i), \tag{3}$$

where $\boldsymbol{C}_i$ includes all children of i. If the pairwise term relies on no image cues but only on relative geometric deformation between i and j, distance transform[8] is applicable to fast maximization of (3); its computational cost is reduced from $O(l^2)$ to $O(l)$. In this formulation, $P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$ consists only of the deformation score, $D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$:

$$P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) = D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) \tag{4}$$

In our model, $D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$[31] is expressed as follows:

$$D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) = \boldsymbol{w}^{i,j} \cdot \psi(\boldsymbol{p}_i, \boldsymbol{p}_j) \quad = \\ \boldsymbol{w}^{i,j} \cdot [x_i - x_j, (x_i - x_j)^2, y_i - y_j, (y_i - y_j)^2]^T, \tag{5}$$

where $\boldsymbol{w}^{i,j}$ denotes a weighted parameter, and $(x_i, y_i) \in \boldsymbol{p}_i$ and $(x_j, y_j) \in \boldsymbol{p}_j$ are the locations of i-th and j-th parts.

The parent j collects the messages from its all children and passes the message to its parent recursively towards a root node.

After all messages are passed recursively toward the root, the pose parameters of the root, $\boldsymbol{p}_{root}$, is determined so that the following score is maximized:

$$M(root) = S^i(\boldsymbol{p}_{root}) + \sum_{c \in \boldsymbol{C}_{root}} M(c, root) \tag{6}$$

The pose parameters of children, which maximize (1), are then determined by backtracking the tree downward.

For pose estimation described above, $S^i(\boldsymbol{p}_i)$ and $P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$ must be trained properly. This training is achieved in advance by using training data where the ground-truth of the configurations of all parts is given. The details of the training step are described in Sec. 5.

(a) input image (b) segmented regions (c) contours

Figure 3. Contour extraction via segmentation by quick shift[29].

4. Parts Connectivity with Discriminatively Trained Local Oriented Contours

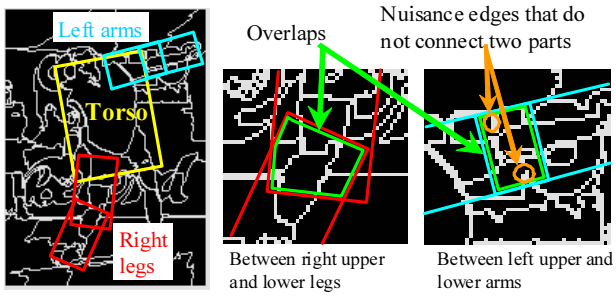
The goal of this work is to embed a contour-based score into a pairwise term, $P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$. Contour features are used less often for representing parts connectivity[20, 25, 22]. Existing contour-based methods for connecting parts have the following problems. First, features are determined based on heuristics, such as parallelism of limb contours[20, 22], with no training data. Second, smooth and long contours are assumed. The length of only smooth and long contours is counted in [22]. As well as the contours, the length of small gaps between them and an angle between bending contours are also regarded as cues in [20]. Entire perimeters of neighboring parts should be extracted in [25]. These heuristics, the assumption of long and smooth contours, and naive thresholding are sensitive to noise and complicated textures.

Our method solves those problems by the ideas below:

Discriminative training: As with appearance training in each part[21, 7, 1], discriminative training with positive and negative samples is employed for contour learning between neighboring parts. This training relieves us from naive thresholding while discriminativity is improved.

Efficient local contours: Unlike the previous works[20, 25, 22], local but informative contours between neighboring parts are evaluated. A local region for evaluation is adaptively determined depending on the relative location of the neighboring parts. The motivation of local evaluation is twofold; to suppress bad effects due to peripheral edges and to reduce computational cost. This adaptive evaluation is the difference also from existing discriminative training methods for each part's appearance, because those methods are achieved with predefined parameters (e.g. a predefined region size and/or any directed contours).

For contour extraction, we tested normalized cuts[23], hierarchical segmentation from globalPb[2], and super-



(a) Contour image (b) Partially magnified images

Figure 4. Selection of contours for evaluating parts connectivity. Contours in the overlap between neighboring parts are evaluated.

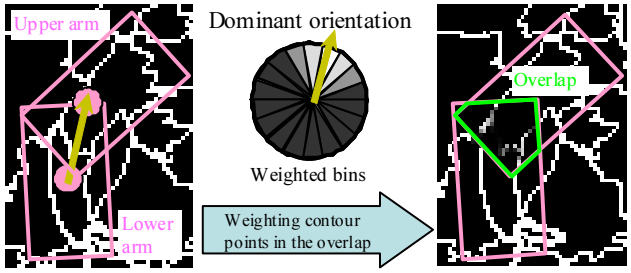


Figure 5. Weighting contour features along the dominant orientation, which is from the child to the parent. Contour points each of whose tangent is toward the dominant orientation are left, while contours oriented to different angles are suppressed. The orientation bins show the weights for this suppression; darker bins are less weighted, where contour points are suppressed strongly. Gray pixels in the overlap show the results of this weighting.

pixelization techniques[14, 16]. The extracted contours should be located on object boundaries with less oversegmentation. It is better that the contours are computed efficiently. For these requirements, we used quick shift[29] (Fig. 3) with the default parameters of its public code.

As described above, our contour-based cues are evaluated in accordance with the relative configuration of the rectangles of neighboring parts. More specifically, the contour cues are extracted from an overlap between two rectangles in a contour image. This is because the strongest cues that represent continuous contours between the parts should be located in where the parts are connected, namely in the overlap, as illustrated in Fig. 4. If longer contours are evaluated beyond the overlap, nuisance edges of other objects and textures might be confused with the contours of the parts of interest. Our strategy is to improve pose estimation accuracy by efficiently employing additional cues (i.e. contour cues between the neighboring parts) with less bad effects. Note that the rectangles of the parts should be relatively large so that they have overlaps between the neighboring parts. For more aggressively extracting contours corresponding the boundary lines of the neighboring parts, we extract contours only that connect the borders of two rectangles in the overlap region; those encircled with orange in Fig. 4 (b) are removed from our contour features.

The orientation of the contour features is also limited in our model. A dominant orientation along which the boundaries of neighboring parts are drawn might be similar to the orientation between child and parent parts. A typical example is observed where upper and lower limbs are oriented in line. Even if the child and parent parts are oriented differently (Fig. 5), a portion of the contours are oriented to the dominant orientation around a certain region with respect to the parts' locations. That dominant orientation is determined so that it is along two joint positions each of which is located in the child or parent parts. The joint is given to wherever a part connects to another. Depending on the tree structure of the articulated model, each part has different number of joints. In an example illustrated in Fig. 2, the head and torso have one and five joints, respectively. In our experiments, all joint positions were given manually[1]

For strongly evaluating contours along the dominant orientation, a tangent at each point along the contours is computed and clustered to orientation bins, which are illustrated in the middle of Fig. 5. The bins are weighted so that they are less weighted as their angle from the dominant orientation gets larger. Each contour point is weighted by the weight of the bin to which it is clustered. By this weighting process, contour points each of whose tangent is different from the dominant orientation are suppressed as shown in the right-hand image of Fig. 5. The weighted contour points are regarded as contour features, which are non-oriented features unlike HOG features; for example, in a window consisting of 5×5 cells, our contour feature is a (5×5)-D vector, while HOG is a $(5 \times 5 \times N^o)$-D vector, where N^o denotes the number of orientation bins.

In our experiments, 18 orientation bins were employed, and the weight of o-th bin, $w(o)$, was given by Gaussian:

$$w(o) = \alpha \exp\left(-\frac{(a_o - a_d)^2}{2\sigma^2}\right), \tag{7}$$

where a_o and a_d denote the o-th bin's and dominant orientations.

The score of the above contour features (denoted by $C^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j)$ for child i and parent j) is expressed by the following form, as with (2) and (5):

$$C^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) = G^{i,j} \cdot \upsilon(I, \boldsymbol{p}_i, \boldsymbol{p}_j), \tag{8}$$

where G^i and $\upsilon(I, \boldsymbol{p}_i, \boldsymbol{p}_j)$ denote the filter between i-th and j-th parts and the contour features extracted from their overlap in image I, respectively. With score (8), the pairwise

[1]One might determine the joint positions by using training data so that they cross the overlap between the respective parts.

function (4) in our model is updated as follows:

$$P^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) = D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) + C^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) \qquad (9)$$

For realizing the score function (8), every contour feature must be extracted from a fixed-size window. This window is called a *contour-evaluation window*. It should contain the overlap between i and j, which is a free-form shape as illustrated by green polygons in Fig. 4 and 5. In our model, the contour-evaluation window is a rectangle whose center is located in the middle point between two joint positions of i and j. The size of the contour-evaluation window can be predefined automatically based on training data; the details of how to determine the size as well as how to optimize the contour filter, $G^{i,j}$, are described in the next section.

5. Implementation Details

5.1. Articulated Model

In our implementation, the articulated model is defined based on a mixture of non-oriented structures proposed by Yang and Ramanan[31]. In their model, each part i has its x-y location and size parameter s as its parameters. Instead of having an orientation parameter θ, the part model consists of a mixture of templates. The training data of i is clustered depending on the relative location of i with respect to its parent part. This clustering is achieved by K-means. The number of clusters in each part is determined empirically. The number of the clusters was 5 or 6 depending on the part in our experiments in accordance with [31]. The ID of the clusters is called a *type*.

Roughly speaking, type selection works in the same manner with selecting θ because the training data is clustered in accordance with the relative orientation between neighboring parts. In addition to this effect, the clustering enables one more advantage. Each type (denoted by t_i of i-th part; $t_i \in \{1, \cdots, K^i\}$) has its own HOG-based appearance filter, F^{t_i}, and observation probability, b^{t_i}. This individual modeling allows each type to specifically represent a subset of largely varying properties (e.g. F^{t_i} and b^{t_i}) of each part depending on its pose. As well as the individual observation probability b^{t_i}, the co-occurrence probability with a type of the parent j, b^{t_i,t_j}, is also employed.

To make the mixture model work robustly to in-plane rotation and foreshortening of limbs, the base model[31] divides physically-rigid parts (e.g. limbs) into several smaller parts. In accordance with the base model, 26 parts were used in our implementation; 2 for the head, 4 for the torso, 10 for the shoulders to the hands, and 10 for the hips to the feet. Each part is shaped like a rectangle. Between those 26 parts, our model has 25 contour-evaluation windows.

The above definition of the parts has high affinity with our contour-based features. The contours are evaluated in the overlap between two rectangles of child and parent parts, as illustrated in Fig. 6. The overlap is always contained by a contour-evaluation window, whose size is equal to a part window, if the joint of each part is located in its center. Figure 6 illustrates examples. In the base model[31], the size of the part window is determined automatically based on the length of parts in training data. By the same rule, the sizes of the part and contour-evaluation windows, which are equal, are determined also in our model.

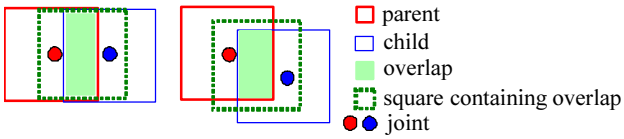


Figure 6. Extraction of contour features between neighboring parts in the mixture of non-oriented structures[31]. In this model, all parts are represented by the same-sized non-oriented rectangles. With this part model, the overlap between the neighboring parts can be expressed easily as a rectangle, which is always contained by a contour-evaluation window, whose size is equal to that of the part rectangle, located in the middle position of the parts.

Finally, the score function (1) is updated with (9), b^{t_i}, and b^{t_i,t_j} as follows:

$$T(\boldsymbol{P}) = \sum_{i \in \boldsymbol{V}} b^{t_i} + \sum_{i,j \in \boldsymbol{E}} b^{t_i,t_j} + \sum_{i \in \boldsymbol{V}} S^i(\boldsymbol{p}_i) + \sum_{i,j \in \boldsymbol{E}} \left(D^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) + C^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) \right), \qquad (10)$$

where $\boldsymbol{p}_i = \{x_i, y_i, s_i, t_i\}$.

5.2. Discriminative Training

The goal of discriminative training is to optimize the model parameters required for computing (10), namely b^{t_i}, b^{t_i,t_j}, F^i in (2), $\boldsymbol{w}^{i,j}$ in (5), and $G^{i,j}$ in (8). For discriminative training, positive and negative examples are collected from training images. The images for the positive examples have the labeled parameters of each part, $\boldsymbol{p}_i$. With these parameters, the positive examples for training, $\phi(I, \boldsymbol{p}_i)$ and $\psi(\boldsymbol{p}_i, \boldsymbol{p}_j)$, are obtained. The negative examples are randomly given from background images with no people.

Let β and $\Phi(I, \boldsymbol{P})$ be a vector consisting of all model parameters of all parts, which would be optimized, and a vector got by concatenating the obtained examples. For establishing a linear classifier $f_\beta(I) = \max_{\boldsymbol{P}} \beta \cdot \Phi(I, \boldsymbol{P})$, the following objective function is minimized:

$$\frac{1}{2}||\beta||^2 + \gamma \sum_i \max\left(0, 1 - z_i f_\beta(\boldsymbol{p}_i)\right), \qquad (11)$$

where γ denotes the weight of regularization, and z_i is -1 or 1 if the i-th example is negative or positive, respectively.

We solve this objective function by quadratic programming, as with the original works[7, 31].

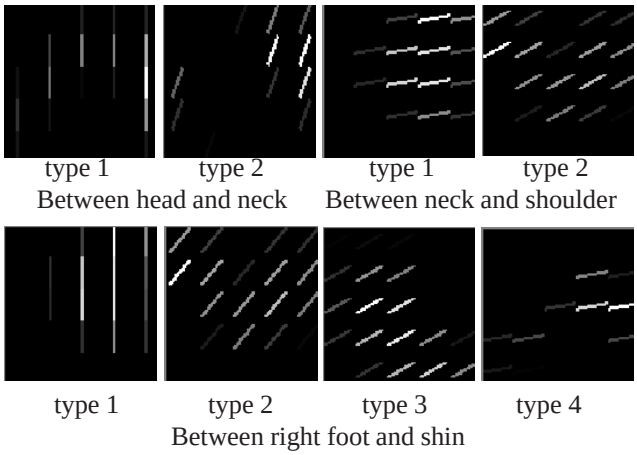


Figure 7. Examples of the filters of discriminatively trained contour features. Each filter consists of 5×5 cells, as well as a HOG filter. But unlike the HOG filter, the contour filter is non-oriented. For visualization purpose, the magnitude of the filter is depicted by a segment slanted along the mean orientation between neighboring parts in training data.

6. Experiments

We tested our proposed model with the Image Parse dataset[19], in which 305 pose-annotated images are included and classified into 100 images for training and other 205 images for evaluation. Negative examples for discriminative training were given from background images in the INRIA Person database[5].

Figure 7 shows examples of trained contour filters. Contour orientations in each pair of the child and parent vary depending on variability of the relative orientation between them. Indeed, as shown in the examples, the contour orientations between the foot and shin varied widely (i.e. 360 degrees) among the types, compared with those between "head and neck" and "neck and shoulder".

In accordance with the implementation in [31], pose estimation results were evaluated as follows. In the dataset, 14 main points are given as the ground-truth. On these 14 points, the centers of the 14 parts out of the 26 parts of our structure model were located. Other 12 parts were located between the 14 main parts. For visual check, the 14 main points and the links between them are shown (Fig. 8 and 9). Quantitative evaluation was achieved also with the 14 main points, which were used for localizing the line segments that define 10 body parts, namely head, torso, and right, left, upper, and lower legs/arms. The estimation accuracy was evaluated by the percentage of correctly localized parts. This quantitative evaluation was done with the code in the BUFFY stickman dataset[9].

Table 1 shows the results of quantitative evaluation. For comparison, the results using the base model with 26 parts[31], (a) in the table, are shown.

Our model consists of 26 parts and 25 contour-evaluation windows. Since our model is based on the base model[31] and intercalates contour-evaluation windows between 26 parts, one might be interested in the difference between our model and the base model[31] with 51 parts. It can be seen that the model with 51 parts could not improve the performance, as shown in Table 1 (b).

From the results of our model without any contour selection in the overlap or weighting contour pixels based on the dominant orientation, (c) in the table, it can be verified that contour selection and weighting contours is effective.

The effect of discriminative training of contour features was also verified. Instead of discriminatively trained features, the mean of extracted contours was used for the contour filter, $G^{i,j}$. The score for contour evaluation, (8), was expressed with the weight of contour (denoted by $\omega^{i,j}$): $C^{i,j}(\boldsymbol{p}_i, \boldsymbol{p}_j) = \omega^{i,j}\left(\bar{v}_{i,j} \cdot v(I, \boldsymbol{p}_i, \boldsymbol{p}_j)\right)$. We tested $\omega^{i,j} \in \{10, 1, 0.5, 0.1, 0.01\}$. The best results with $\omega^{i,j} = 0.01$, (d) in the table, were worse than those of our model, (e).

Figure 8 shows several examples where our method got better results rather than the base model[31]. It can be seen that, with our model, if clear contours that connect "the torso and upper limbs" and "the upper and lower limbs" are observed, the parts could be localized correctly without being disturbed by other clutters, whose appearance might be similar to that of a part.

Three examples of big failures with our model are shown in Fig. 9. In (i), the left lower arm and the right leg were mislocalized. The right leg is clearly observed with no occlusion, and the left lower arm is occluded but its boundary with the upper arm is observed. It might be possible to localize these parts correctly by improving our model. On the other hand, for correctly localizing the lower arms in (ii), richer appearance cues might be required due to severe background clutter. While all parts in (iii) are clearly observed, pose estimation was failed probably because the pose was significantly different from any of those in training images. To resolve this failure, the effects of pose prior given by b^{t_i} and b^{t_i,t_j} should be reconsidered; pose prior might be overfitted to a small amount of training data. One might try to resolve this problem by employing more training data for covering a huge variety of possible poses.

For verifying the possibility of improving the performance with more training data, we trained the models with 200 images (1st to 200th images in the Image Parse dataset). Table 2 shows the results with 100 and 200 training images. For fair comparison, the same set of test images (201st to 305th images in the dataset) were used. Contrary to expectation, the performance was not improved. This result gives an insight into requirements of other cues, much more training data (> 1000 images) with richer attributes[4], etc. Collecting a huge data would cause requirement for processing inaccurate training data[12].

Model	Head	Torso	Upper legs	Lower legs	Upper arms	Lower arms	Total
(a) Mixtures[31] with 26 parts	93.2	97.6	83.9	75.1	72.0	48.3	74.9
(b) Mixtures[31] with 51 parts	95.1	100	83.9	76.1	72.7	42.9	74.6
(c) Our model with no weights	58.5	92.2	52.7	41.5	43.4	27.3	48.1
(d) Our model with no contour training	95.6	99.0	85.4	75.1	73.2	46.8	75.6
(e) Our model	98.5	100	89.8	79.0	77.6	51.2	79.4

Table 1. Comparative results of the percentages of correctly localized parts in pose estimation. (a) mixture model of non-oriented 26 parts, (b) mixture model of non-oriented 51 parts, (c) our model without weighting contour points based on the dominant orientation, (d) our model using a weighted score of contour features instead of their discriminative training, and (e) our model.

Training images	Head	Torso	Upper legs	Lower legs	Upper arms	Lower arms	Total
(a) 100 images	96.2	100.0	88.6	79.0	76.2	49.5	78.3
(b) 200 images	95.2	99.0	86.7	79.0	75.3	50.5	77.7

Table 2. Model training was achieved with a different number of training images. (a) and (b) were trained with "the 1st to 100-th images" and "the 1st to 200-th images" in the Parse Image dataset. All models were evaluated with the 201st to 305-th images.

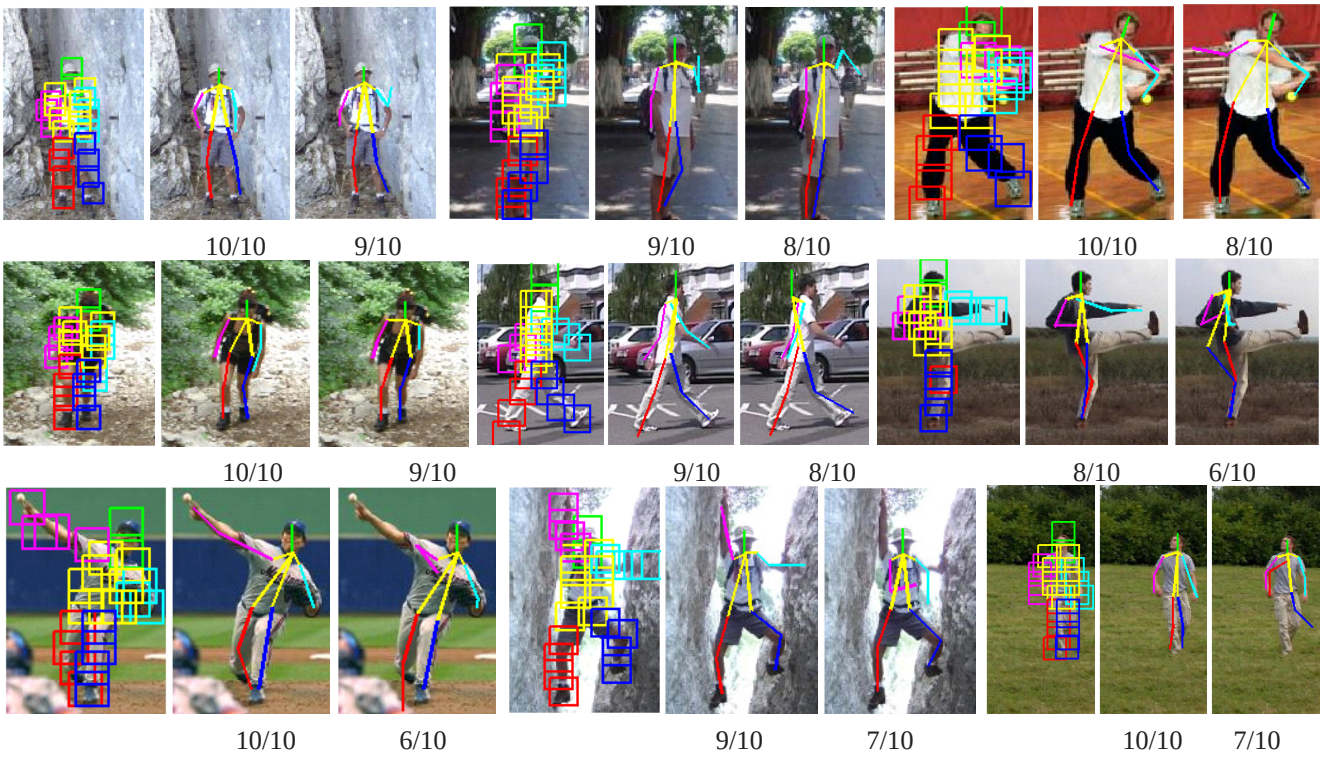
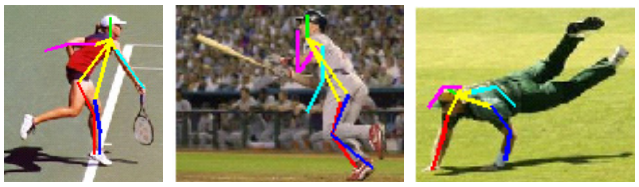


Figure 8. Pose estimation results. For each test image, three results are shown: (Left) 26 part-rectangles estimated by our method, (Middle) 14 main points extracted from the estimated 26 rectangles, and the links connecting these points, (Right) results estimated by the method without appearance cues between neighboring parts[31]. The number of correctly localized parts is shown under each result.

For more comparison, estimation accuracy was tested with the BUFFY stickemen dataset[9], as shown in Table 3. The proposed method had less impacts in the BUFFY dataset compared with the Parse dataset. This might be because many images in the BUFFY have low contrast that makes contour extraction difficult. This problem might be alleviated by adaptive thresholding.

7. Concluding Remarks

This paper proposed contour-based features for connecting neighboring parts in articulated pose estimation. The features are well localized and discriminatively trained. The feature localization suppresses bad effects of nuisance edges and reduces computational cost. Discriminative training improves connectivity of neighboring parts.



(i) (ii) (iii)

Figure 9. Examples of estimation failure.

Model	Head	Torso	U-arms	L-arms	Total
(a) [22]	96.2	100.0	95.3	63.0	85.5
(b) [31]	99.6	100.0	96.6	70.9	89.1
(c) Ours	98.9	100.0	97.5	73.9	90.3

Table 3. Comparative results with the BUFFY stickmen dataset[9]. (a) cascaded model[22], (b) mixture model of non-oriented 26 parts[31], and (c) our model.

Future work of our model includes 1) more optimized featurization of contours in terms of the size and location of the contour-evaluation window relative to parts and 2) other good appearance cues for connecting neighboring parts.

The codes of the base model[31] and quick shift[29] were given by their respective authors.

References

[1] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *CVPR*, 2009. 2, 3

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, 2011. 3

[3] M. Bergtholdt, J. H. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *International Journal of Computer Vision*, 87(1-2):93–117, 2010. 2

[4] L. D. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009. 2, 6

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2, 3, 6

[6] P. F. Felzenszwalb, R. B. Girshick, and D. A. McAllester. Cascade object detection with deformable part models. In *CVPR*, 2010. 2

[7] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2010. 2, 3, 5

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005. 1, 2, 3

[9] V. Ferrari, M. J. Marín-Jiménez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *CVPR*, 2008. 2, 6, 7, 8

[10] M. Isard. Pampas: Real-valued graphical models for computer vision. In *CVPR*, 2003. 2

[11] H. Jiang and D. R. Martin. Global pose estimation using non-tree models. In *CVPR*, 2008. 2

[12] S. Johnson and M. Everingham. Learning effective human pose estimation from inaccurate annotation. In *CVPR*, 2011. 2, 6

[13] J. H. Kappes, S. Schmidt, and C. Schnörr. Mrf inference by *k*-fan decomposition and tight lagrangian relaxation. In *ECCV*, 2010. 2

[14] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(12):2290–2297, 2009. 4

[15] Y. Li, L. Gu, and T. Kanade. A robust shape model for multi-view car alignment. In *CVPR*, 2009. 2

[16] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa. Entropy rate superpixel segmentation. In *CVPR*, 2011. 4

[17] G. Mori and J. Malik. Recovering 3d human body configurations using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(7):1052–1062, 2006. 2

[18] G. Mori, X. Ren, A. A. Efros, and J. Malik. Recovering human body configurations: Combining segmentation and recognition. In *CVPR*, 2004. 2

[19] D. Ramanan. Learning to parse images of articulated bodies. In *NIPS*, 2006. 2, 6

[20] X. Ren, A. C. Berg, and J. Malik. Recovering human body configurations using pairwise constraints between parts. In *ICCV*, 2005. 2, 3

[21] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *ECCV*, 2002. 2, 3

[22] B. Sapp, A. Toshev, and B. Taskar. Cascaded models for articulated pose estimation. In *ECCV*, 2010. 2, 3, 8

[23] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000. 3

[24] V. K. Singh, R. Nevatia, and C. Huang. Efficient inference with multiple heterogeneous part detectors for human pose estimation. In *ECCV*, 2010. 2

[25] P. Srinivasan and J. Shi. Bottom-up recognition and parsing of the human body. In *CVPR*, 2007. 3

[26] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *CVPR*, 2003. 2

[27] T.-P. Tian and S. Sclaroff. Fast globally optimal 2d human detection with loopy graph models. In *CVPR*, 2010. 2

[28] D. Tran and D. Forsyth. Improved human parsing with a full relational model. In *ECCV*, 2010. 2

[29] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *ECCV*, 2008. 3, 4, 8

[30] H. Wang and D. Koller. Multi-level inference by relaxed dual decomposition for human pose segmentation. In *CVPR*, 2011. 2

[31] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 2, 3, 5, 6, 7, 8

[32] J. Zhang, J. Luo, R. T. Collins, and Y. Liu. Body localization in still images using hierarchical models and hybrid search. In *CVPR*, 2006. 2