Recognizing Human Postures and Poses in Monocular Still Images

J.P. Wachs, D. Goshorn, and M. Kölsch
MOVES Institute/CS Department, Naval Postgraduate School, Monterey, CA, USA

Abstract – *In this paper, person detection with simultaneous or subsequent human body posture recognition is achieved using parts-based models, since the search space for typical poses is much smaller than the kinematics space. Posture recovery is carried out by detecting the human body, its posture and orientation at the same time. Since features of different human postures can be expected to have some shared subspace against the non-person class, detection and classification simultaneously is tenable. Contrary to many related efforts, we focus on postures that cannot be easily distinguished after segmentation by their aspect ratio or silhouette, but rather require a texture-based feature vector. The approaches presented do not rely on explicit models nor on labeling individual body parts. Both the detection and classification are performed in one pass on the image, where the score of the detection is an ensemble of votes from parts patches.*

Keywords: posture recognition, part-based detectors, pose detection, multi-class detectors, Adaboost.

1 Introduction

Detection and recognition of human postures from images attracts an increasing amount of research in the last years. Surveillance, human-machine interaction, multimedia retrieval, health-care and biometrics are only a few disciplines that motivates original contributions in the area of machine vision. These works can be divided into three dimensional (3D) and two dimensional (2D) human posture recognition. 3D view-based human posture recognition requires the collection of a large set of independent 2D view images of the scene, which is computationally expensive and produces a huge search space. Moreover, installation, calibration, object matching, switching, data fusion and occlusion are noted in [1] as the main problems in multiple camera systems. Alternatively, the work presented in [2], illustrates how 3D human posture can be recovered from still images efficiently and accurately, which goes hand in hand with the requirement of low cost monocular solutions relying heavily on simple visual features obtained from a single view.

Many approaches have been proposed for human posture recognition. Most of their recognition accuracy is affected by the high variability that exists within the same postures performed by different people.

3D human body pose is recovered from monocular image sequences in [3] by applying non-linear regression on histogram-of-shape context descriptor vectors. This system was validated with human walking sequences on a low-clutter background. In [4] a system was developed to monitor human-behavior for safety purposes. Four main postures in each of three views (frontal, left-headed, and right-headed) were classified: standing, crouching, sitting and laying. The classification was performed by using a Bayesian framework utilizing features extracted from projection histograms of the silhouette. Projection histograms [5] use an HMM for classification and a multi-camera setup to overcome occlusions. Four main postures were detected: standing, crawling, sitting, and lying. The authors rely strongly on a successful segmentation in the preprocessing stage since the features are extracted from the human silhouette blobs. Juang et al.'s work [6] also requires a successful segmentation for their visual surveillance system to recognize four postures: standing, bending, sitting, and lying. The features are obtained from the DFT coefficients from projected histograms, similar to [4-5] and the classification is done through a neural fuzzy inference network. In [8], 2D images are collected from different view angles and Fourier descriptors are extracted from the contours. Classification is obtained using aspects-graph representation to recognize eight human postures, including standing, kneeling, sitting and laying down. Image-matching on successive convexification and linear programming [9] can successfully recognizing human postures in cluttered images and videos Yoga poses, skating and baseball postures are recognized using this approach.

In summary, there are two main approaches for recovering human pose from images: model-based approach (or direct approach) and the learning based approach (or indirect approach) [3]. Model-based approaches assume that a parametric body model is known. Through incrementally predicting and updating pose variables a cost function representing the pose is optimized. Learning-based approaches do not require a detailed human body model. The model is learned by training examples representing typical human poses and followed by search and comparison, where the new poses are interpolated.

In our work we adopt the learning based approach, and, in particular, a parts-based approach, where the parts are stored in codebooks and later detected in promising regions (for example found with interest point detectors) rather than searching in all the possible subwindows of an image. The detection is based on casting votes for the object center from the parts matched. This approach was recently adopted for articulated objects detection such as pedestrians and showed very good performances [10-14]. We implemented the multi-class approach presented in [15] for marine posture recognition. This will be integrated with into the BASE-IT system, an intelligent methodology for marine's training evaluation using behavior analysis [16].

As opposed to [15] the multiview appearance of each of the postures that we consider is very similar (e.j. marine standing back and front view) and hence the classification task is extremely more difficult.

We discuss the parts-based method in Section 2 for a single class and for a multiclass problem. Our dataset and a detailed description of the experiments and results are presented in Section 3. Section 4 suggests further directions of improvements and concludes the paper.

2 Parts-based object detection

In this section, we briefly review our approach of object detection using single and multi-class detectors. First, we will describe the feature extraction process from patches (or parts) and then we will describe the basic and shared classifiers.

2.1 Dictionary of parts

The dictionary is built from features (patches) extracted from a set of eight images per class, similar to [12]. For each image we apply the following operations: 2D convolution using each filter from a bank of four filters: a delta function, x and y derivatives and a Gaussian. Following the convolutions, 20 patches are selected in *x,y* locations randomly, lying inside the annotated silhouette of the object and over interest points, found by a simple canny edge detector. In those locations *x,y* patches are extracted in all the filtered images and also grayscale normalized. The patches' sizes are described by the triplet {width, height and depth} where the first two are symmetric are selected randomly between 9x9 to 25x25, and the depth is equal to the number of filters, 4. Prior to extracting the patches, the objects are normalized to a standard scale of 128x48 for marines standing and to 64x48 for marines kneeling. Each patch is also associated with the place from where it was extracted, relative to the center of the object. This location information is stored as a binary mask centered at the object center. This mask, in turn, is decomposed in two vectors containing the *x,y* offset distances respectively $\{l_x, l_y\}$, after applying a blurred delta function to them. The reason for this it is that is faster to compute cross correlation using vectors than matrices. Hence, each entry *i* in the dictionary has the form v_i={*filter, patch,* l_x, l_y, *image no.*}. A total of 640 entries per class were obtained using the procedure which is depicted in Figure 1.

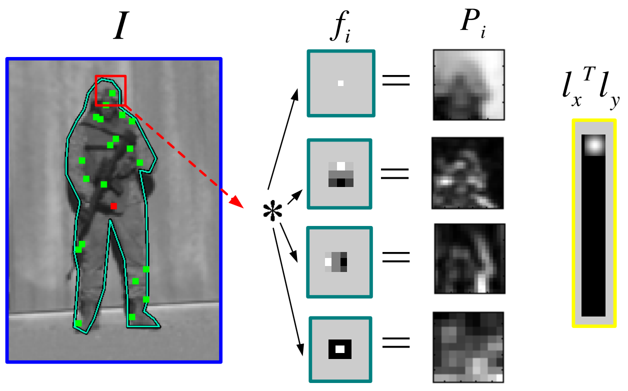


Figure 1. Dictionary entries: patches selected randomly (on the left image) are convolved with a bank of filters. The position of the patches is represented by the location matrix (right). Since the red patch is at almost the same horizontal position and at the top, relative to the center, the position matrix has a bright spot.

2.2 Computing the training vectors

A set of training images including each of the eight objects that we are interested detecting is collected for the training phase of the system. From each image a few feature vectors are obtained using the following method:

1. Scale all the images in the training set so the objects are enclosed in a bounding box of 128x48 and 64x48 for standing and kneeling respectively, and the images are not larger than 200x200.

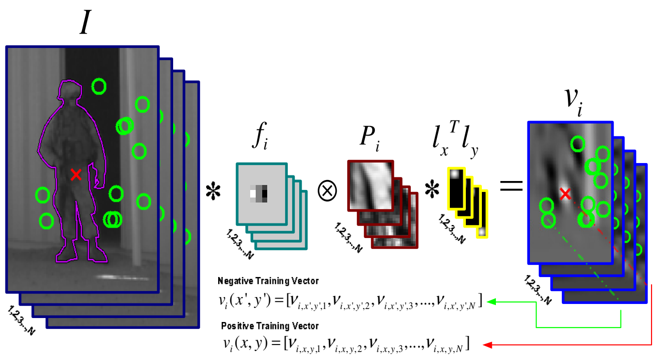
2. For each image *j* normalized in scale, each entry *i* of the dictionary is applied to it in the following way. The image is convolved with the filter in entry *i*, and convolved again with a Guassian to smooth the response. Next, it is cross-correlated with the patch in entry *i*, yielding a strong response where this patch appears in the filtered image. Finally, the 1D filters l_x and l_y are applied to the cross-correlated image, effectively "voting" for the object center. This is summarized in Eq. 1:

$$v_i(x,y) = \left[\left(I * f_i \right) \otimes P_i \right] * l_x^T l_y \tag{1}$$

Where $*$ is the convolution operator, $\otimes$ is the normalized cross correlation operator, $v_i(x,y)$ is the feature vector entry i, f is a filter, P is a patch, and l_x and l_y are the x,y location vectors with respect to the center of the image respectively.

3. For each image in step 2, we extract feature vectors $v(x,y)$. A positive sample vector is obtained by retrieving v at the x,y coordinates in the center of the object. The negative training samples were a subset of 20 vectors retrieved from x,y locations that had a high response to (1).

Each of these vectors is accompanied with a class label (1 to 8) and -1 for negative samples. Given 25 images per class, we obtain a training set of 4000 negative and 200 positive samples, each with 640 features, see Figure 2.



I
f_i
P_i
$l_x^T l_y$
v_i
1,2,3,...,N
Negative Training Vector
$v_i(x',y') = [v_{i,x',y',1}, v_{i,x',y',2}, v_{i,x',y',3}, \ldots, v_{i,x',y',N}]$
Positive Training Vector
$v_i(x,y) = [v_{i,x,y,1}, v_{i,x,y,2}, v_{i,x,y,3}, \ldots, v_{i,x,y,N}]$

Figure 2. Positive and negative vector set creation using the dictionary entries and sampling the center out of the silhouette points. Each sampled point, is a vector, where an entry j in the vector represents the number of votes assigned by patch P_i.

2.3 Multiclass Adaboost with shared features

In this section we briefly describe the joint boosting algorithm used for multi-class multi-view object detection. For a more detailed discussion, refer to [15].

A boosting algorithm is an additive model where weak learners are sequentially added to form a strong classifier. For the multiclass case, the strong learner is defined as:

$$H(v,c) = \sum_{m=1}^{M} h_m(v,c) \tag{2}$$

Where v is the input feature vector, M is the number of boosting iterations, c is a specific class and $H(v,c)=\log P(zc=1|v)/P(zc = -1|v)$ is the logistic function where z is the membership label (±1). When the expectation is replaced by an average over the training data, the cost function can be written as:

$$J_{wse} = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c \left(z_i^c - h_m(v_i, c) \right)^2 \tag{3}$$

Where N is the number of training vectors, w_i^c are the weights for sample i and for class c, z_i^c is the membership label for sample i for class c (±1). The weak shared learner, also called, regression "stump" is defined for the multiclass in (4):

$$h_m(v,c) = \begin{cases} a_S & \text{if } v_i^f > \theta \text{ and } c \in S(n) \\ b_S & \text{if } v_i^f \le \theta \text{ and } c \in S(n) \\ k_S^c & \text{if } c \notin S(n) \end{cases} \tag{4}$$

where v_f is the component f^th from the vector v, θ is a threshold, δ is the indicator function, a_S and b_S are regression parameters. $S(n)$ is a subset of the classes labels. Each round of boosting consists of selecting the shared "stump" and the shared feature f that minimizes (3), from the subset of classes $S(n)$, in the following stated procedure: Pick a subset of classes $S(n)$. Search all the components f of the feature vector v, for each component, search over all the discrete values of θ and for each couple $\{f, \theta\}$, find the optimal regression parameters a_S and b_S using (5-7). Finally, select $\{f, \theta, a_S, b_S\}$ that minimizes (3).

$$a_S(f,\theta) = \frac{\sum_{c\in S(n)} \sum_i w_i^c z_i^c \delta\left(v_i^f > \theta\right)}{\sum_{c\in S(n)} \sum_i w_i^c \delta\left(v_i^f > \theta\right)} \tag{5}$$

$$b_S(f,\theta) = \frac{\sum_{c\in S(n)} \sum_i w_i^c z_i^c \delta\left(v_i^f \le \theta\right)}{\sum_{c\in S(n)} \sum_i w_i^c \delta\left(v_i^f \le \theta\right)} \tag{6}$$

$$k^c = \frac{\sum_i w_i^c z_i^c}{\sum_i w_i^c} \tag{7}$$

Therefore a shared weak learner is associated with a set of 6 parameters $\{f, \theta, a_S, b_S, k_c, S_n\}$ of the subset of classes selected. It is more efficient to keep a pointer to the entry in the dictionary from where f was obtained rather than keeping the whole feature vector. This will also provides us with the patch, filter and location vectors entries in the dictionary which will be used for the detection stage. This new weak learner is added to the previous accumulated learner, for each training example: $H(v_i, c) = H(v_i, c) + h_m(v_i, c)$ where h_m is computed for the optimal subset of classes. The optimal subset of classes is the one that minimize the misclassification error by selecting a feature shared by those classes. Finally, the chain the chain of weak learners is stored in the accumulated learner.

2.4 Detection

To detect an object of class *c* in a test image we need to compute the score for every pixel in the image, provided by the strong classifier *H(v,c)* evaluated in all the pixels. If the score exceeds some threshold the object is detected. In order to calculate *H(v,c)* we use the following procedure.

We find all the shared weak learners that shares class *c*, and for each sharing weak learner do:

1. Obtain the 4-tuple $\{f, \theta, a_S, b_S\}$ from the weak learner. Since *f* is associated with an entry in the dictionary, we retrieve the corresponding filter, patch and vectors L_x, L_y from the dictionary, and apply them to the test image using (1).
2. Calculate $h_m(v)= a\delta(v_f > \theta)+b$ where V_f is the image obtained in the previous step.

Finally add up all the weak learners together. Each weak learner votes for the center of the object that we are searching for, and it is expressed by a grayscale image obtained in step 2. The accumulated image will have bright pixels where the weak learners "agreed" about the center of the object in the "voting space". A maxima in the accumulated image indicates the probability to find the object in that location.

Each strong detector of a different class outputs an accumulated image. Thus, it is possible that more than one strong detector will vote for the same (or very close) pixel coordinates. This situation is not rare since some postures are very similar. To solve this conflict, peaks that are closer than a given radius are clustered together, and the resulting class of the detection is the one from the class with the highest maxima.

3 Experiments and results

We applied the multi-class detector to the problem of posture classification and orientation extraction to images including marines performing eight body configurations (two postures in four orientations each), captured in an actual training environment. For this experiment, we trained a person detector with the traditional Viola & Jones method [17] and Leibe et al. [13] codebook based approach.

3.1 Performance

In order to compare the performance of both detectors we consider two tasks: (a) object detection and, (b) detection and class recognition. In the first case we are interested in finding the probability of a marine's presence in general, that is, in any pose or posture. The second experiment determines the ability to distinguish between the configurations in addition to finding the exact location of the marines.

Testing and training relied on our online database of images which is an extended version of the MIT-CSAIL online annotation tool and database LabelMe [7] but allowing azimuth orientation annotation for the objects. We manually annotated 4166 images of marines performing several exercises, from which we selected eight different classes: two postures (standing and kneeling) and four orientations each, based on the torso (frontal, oriented left, right or away from the camera). For the dictionary creation, eight samples per class were used, for training 25 images and for testing another 25 per class. The images were resized to 128 x 48 for standing, and for kneeling to 64 x 48 and then images were cropped to a size of 200 x 200 for the training set and 256 x 256 for the testing set.

Our first experiment consisted of object location detection. We used a straightforward approach of running the multiclass detector on the image and considered any class detection falling into the "true" annotated bounding box as a hit. For the Viola-Jones detector, we evaluated the common stages of the cascade tree and skipped the branches. Each detection has a score assigned, which represents the votes casted for that detection. We normalize each detection score to a probability by dividing each score by the maximum score detected in the testing set, hence the maximum score is $s_{max}(i, I)=P(O = 1|.)$ where *i* is the location in image *I* where the best score was obtained. By varying the threshold of the scores (between 0-1), we obtain a ROC curve (Figure 3). False alarms are counted per test area / per image which had an average of 30000 test areas.

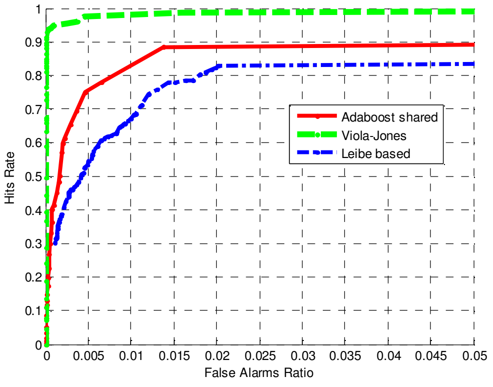


Figure 3. ROC Plot for the marine location detector

The second experiment shows how good the multi-class detector can deal with the two specific postures and four orientations in the images.

A true hit is obtained if the class voting for the detection corresponds to the true annotated bounding box, otherwise the target was missed. Other detections from the remaining classes (regardless if they detected the marine or not) are considered false alarms. We adopt the precision-recall metric (PR) rather than ROC since the former is defined for binary detection problems and not multi-class, and is not a solution independent metric when calculating the false alarm rate.
The PR points are obtained by changing the score threshold, as done in the previous experiment, see Figure 3. Given that TP is the number of true hits, nP is the number of objects to be detected, FP is the false alarms, then recall is defined as TP/nP and precision is TP/(TP+FP) . The performance of the PR is expressed by the F1 score = (2*recall*precision)/(recall+precision). The F1 score is summarized in Table 1.

Table 1. F1 score for posture detection

	Class							
Method	1	2	3	4	5	6	7	8
Sharing	0.509	0.298	0.4	0.05	0.697	0.5	0.582	0.755
Leibe's based	0.274	0.2	0.129	0.221	0.211	0.151	0.217	0.219

The performance of posture recognition is summarized in Table 2.

Table 2. Confusion matrix for 8 posture categories: (a) Multi-class Sharing; (b) Leibe's based, and (c) Viola-Jones

Given class	*Assigned Class*							
	1	2	3	4	5	6	7	8
Standing 0	19	4	5	6	0	0	1	0
Standing 90	4	11	8	6	0	1	0	0
Standing 180	9	4	12	4	0	0	0	0
Standing 270	3	5	0	20	0	0	0	0
Kneeling 0	0	0	0	0	16	2	2	2
Kneeling 90	0	0	0	0	2	17	3	2
Kneeling 180	0	0	0	0	0	1	26	0
Kneeling 270	0	0	0	0	0	1	2	20

(a)

Given class	*Assigned Class*							
	1	2	3	4	5	6	7	8
Standing 0	17	10	6	12	9	0	5	1
Standing 90	15	18	3	12	7	1	8	1
Standing 180	11	15	6	11	7	0	5	0
Standing 270	5	12	8	17	2	1	9	1
Kneeling 0	0	0	0	0	16	1	1	5
Kneeling 90	0	0	0	0	6	8	4	7
Kneeling 180	0	0	0	0	5	3	10	5
Kneeling 270	0	0	0	0	8	5	1	14

(b)

Given class	*Assigned Class*							
	1	2	3	4	5	6	7	8
Standing 0	12	2	0	2	7	0	3	0
Standing 90	0	4	2	2	2	1	1	1
Standing 180	3	1	6	0	5	0	3	0
Standing 270	0	3	0	3	0	0	0	1
Kneeling 0	0	1	0	0	19	7	6	1
Kneeling 90	0	2	0	5	4	21	5	1
Kneeling 180	1	0	0	0	8	1	4	6
Kneeling 270	1	0	0	2	5	3	3	6

(c)

The confusion matrices shows the number of postures classified correctly, the (diagonal values), and the confused postures (off-diagonal). See Figure 7 for examples of typical detections and posture recognitions.

The multi-class Viola-Jones detector does not perform a non-maximum suppression but rather accepts all matches for an area. Hence the total number of detections can be higher than the number of areas supplied.

3.2 Shared Classes

To gain insights about the effect of the features selected, of those shared and also the classifier performance, we study the distribution of features selected by the strong detector. In order to discuss the concept of sharing architecture, let defines a sharing topology as a binary number B_i, where each digit represent a class. Thus, for a given binary number having K being "ones", means that K classes share a feature. The decimal representation of this value is $D(B_i)$. For example, the sharing topology B_i = 00001101 means that classes 1,3 and 4 are shared by the weak shared learner i.

The distribution of shared classes can be obtained by observing the number of times that specific subsets of weak detectors share a feature in our strong classifier. Figure 5 shows that the seven top most popular weak detectors do not share classes, but the last two do share classes: $D(B_i)$= 1,4,8,32,2,64,16,144,6 used 11,10,9,9,7 ,7,6,6,4 times, respectively. The most popular sharing occurs for $D(B_i)$=144, which is B_i= 10010000 (with 6 occurrences) meaning that classes 5 and 8 are shared (MSB is in the left). Since shared features can generalize better on similar data, this could explain the reason that both the F1 scores, for class 5 and 8 are the higher than all the others (~0.7 and 0.75).

In addition, we present the proportion of classes that have features selected from the dictionary (to which classes the patches selected belong to) by the weak detectors in Figure 6. The fact that the class owing the most popular feature is '8' may explain also the reason that class '8' has the highest F1 score. Even though the dictionary has an equal number of entries (patches) from the different classes, the strong detector "picks" the patches that best separate the classes feature-space.

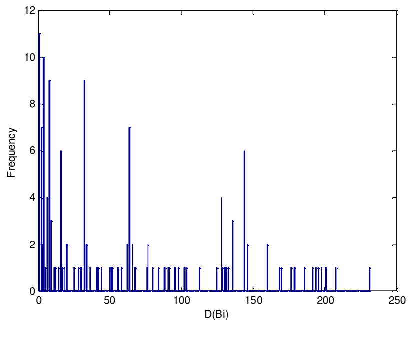


Figure 5. Distribution of weak shared learners within the strong classifier. D(Bi) is the binary encoded sharing topology.

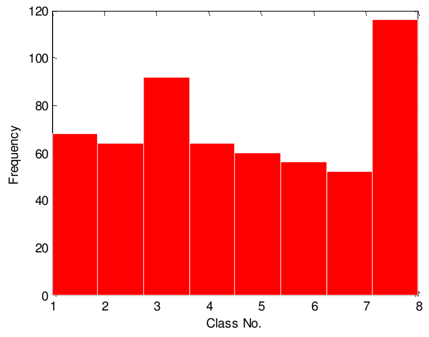


Figure 6. Distribution of the classes having features selected by the strong detector

4 Conclusions

To conclude, we applied three approaches to articulated (human) body pose and posture recognition with the goal of providing input to a behavior analysis system. Our results of the ROC curves show that the Viola-Jones/"complete" method (non-parts based) performs well on the detection task only, but performs worse on the more fine-grained posture/pose recognition than the parts-based methods. The results support the hypothesis that feature sharing contributes to posture/pose recognition during training and testing, and that the patches capitalize on the commonality between postures/poses without loosing their characteristics. This motivates future work to use the Viola-Jones method for detection only, and the parts-based approach for posture recognition only on detected area. This will not only increase accuracy of the posture recognition, but will speed up the whole process.

Results indicate the multi-class sharing method achieves higher posture recognition rates than the Leibe-based method (see Section 3.2 and Table 2(a)-(b)). The multi-class sharing method selects dictionary entries based on the amount of common features between classes, while the Leibe-based approach does not take this into account. However, the Leibe-based approach is significantly faster, since it requires less convolutions. The detection and recognition performance is encouraging and we will extend our dataset to a larger number of classes. With additional temporal processing, our methods are expected to perform sufficiently well for behavior analysis.

In future work, we are interested in combining the two parts-based approaches presented in this paper to yield a faster parts-based method which takes advantage of multi-class feature sharing.

5 References

[1] A. W. Hu, T. Tan, L.Wang, S. Maybank, "A Survey on Visual Surveillance of Object Motion and Behaviours", IEEE SMC-C 34(3):334–352, 2004.

[2] T.B. Moeslund, E. Granum: "A Survey of Computer Vision-Based Human Motion Capture", Computer Vision and Image Understanding, 81:231-268, 2001.

[3] A. Agarwal and B. Triggs "Recovering 3D Human Pose from Monocular Images", IEEE Trans. Pattern Anal. Mach. Intell., 28:1:44-58, 2006.

[4] Cucchiara, R., Grana, C., Prati, A, Vezzani, R."Probabilistic Posture Classification for Human-Behavior Analysis", SMC-A(35), No. 1, January 2005, pp. 42-54.

[5] Cucchiara, R. Prati, A. Vezzani, R. "Posture classification in a multi-camera indoor environment", IEEE Intl Conf on Image Proc, ICIP 2005, I- 725-8, 2005.

[6] Juang, C.F., Chang, C.M., "Human Body Posture Classification by a Neural Fuzzy Network and Home Care System Application", IEEE SMC-A (37), No. 6, November 2007, pp. 984-994.

[7] B. Russell, A. Torralba, K. Murphy, W. T. Freeman. "LabelMe: a database and web-based tool for image annotation". Intl Journal of Computer Vision, 2007.

[8] J.-S. Hu T.M. Su P.C. Li."3-D Human Posture Recognition System Using 2-D Shape Features". 2007 IEEE International Conference on Robotics and Automation. April 2007. pp. 3933-3938.

[9] Hao Jiang, Ze-Nian Li, and Mark S. Drew. "Recognizing Posture in Pictures with Successive Convexification and Linear Programming". IEEE Multimedia, 2007.

[10] G. Bouchard and B. Triggs. A hierarchical part-based model for visual object categorization. In CVPR, 2005.

[11] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In CVPR, 2005.

[12] K. Murphy, A. Torralba, D. Eaton, W. T. Freeman. "Object detection and localization using local and global features". Lecture Notes in Computer Science. Sicily workshop on object recognition, 2005.

[13] B.Leibe, E. Seemann, and B. Schiele: Pedestrian Detection in Crowded Scenes. CVPR (1) 2005: 878-885.

[14] P. Felzenszwalb, D. McAllester, D. Ramanan. "A Discriminatively Trained, Multiscale, Deformable Part Model". Proceedings of the IEEE CVPR 2008.

[15] S A. Torralba, K. P. Murphy and W. T. Freeman. "Sharing visual features for multiclass and multiview object detection" IEEE PAMI, 29:5, pp. 854-869, 2007.

[16] BASE-IT: Behavior Analysis and Synthesis for Intelligent Training. On-line: http://www.movesinstitute.org/base-it/index.html

[17] P. Viola, M. Jones. "Robust Real-time Object Detection," Second Intl Workshop on Statistical and Comp. theories of Vision, Vancouver, July 2001

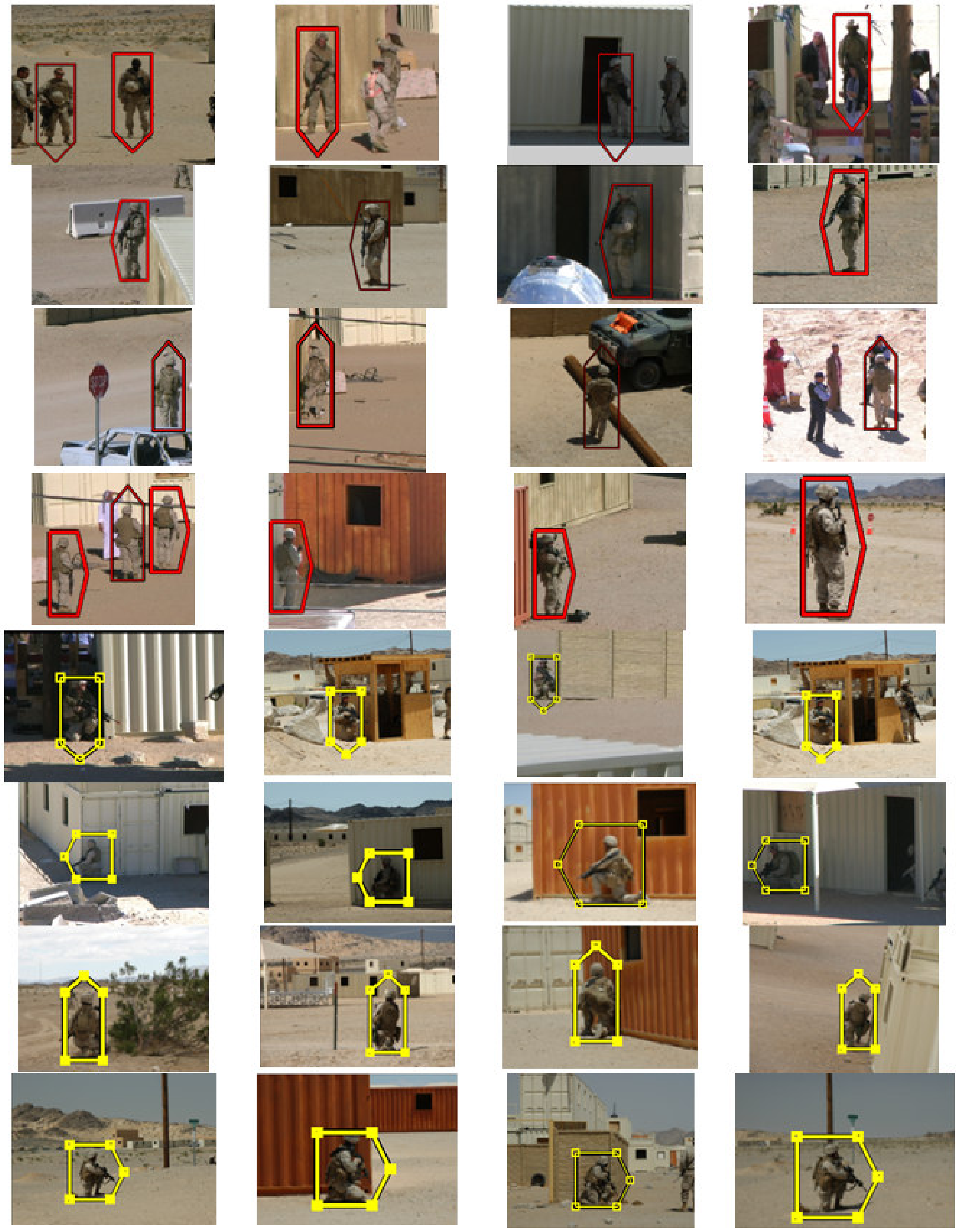


Figure 7. Examples of detections: Each row of images shows the detection of one specific class. The first four classes are ordered from top to bottom as following: standing 0, 90, 180, 270 degrees torso. The last four classes are, from top to bottom: kneeling 0, 90, 180, 270 degrees torso. The color of the bounding box is red for standing and yellow for kneeling. The orientation is expressed by the direction of the arrow within the bounding box. The boldness of the bounding box is proportional to the score of the detection.