Pedestrian Detection and Tracking

Vimal Sharma,Shikhar Sharma
Supervisor : Dr Amitabha Mukerjee
{ svimal,shikhars,amit } @cse.iitk.ac.in

April 11, 2012

Abstract

We use deformable part based models [6] of human body to detect[4] and track pedestrians in a video. These models must be robust and capable of detecting pedestrians in a wide variety of poses/clothing and even if some of their body parts are occluded.

Related Work

A significant amount of work on part based deformable models has been done in past. Earlier works [9] in part based models have used boosted classifiers with weakly classified multiple local part detectors. In [5, 7] , description of object in terms of parts and subparts is represented by grammar productions , where the non-terminals represent objects and terminals represent the appearance parameters.

Part Based Deformable Models

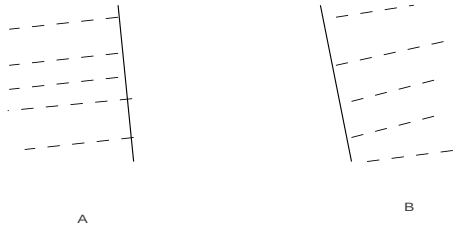
The model represents the body as a deformable configuration of individual parts which are in turn are modelled separately in a recursive manner. One way to visualize the model is a configuration of body parts interconnected by springs. The spring like connections allow for the variations in relative positions of parts with respect to each other. The amount of deformation in the springs acts as penalty(deformation cost).
Matching of such a model to an image can be described mathematically as [6]:
Let $G = (V, E)$ be an undirected graph where the vertex v_i represents the

center of i^{th} part and edge (v_i, v_j) denotes that i^{th} and j^{th} part are connected. In an image,suppose the configuration $L = (l_1.....l_n)$ denotes that i^{th} part is at location l_i. Let $m_i(l_i)$ represent the amount of mismatch when i^{th} part is placed at location l_i. Further, let $d_{ij}(l_i, l_j)$ be the function which gives the relative deformation cost when parts i and j are placed at l_i and l_j respectively. The best match is the one which minimizes

$$L_{opt} = argmin_L(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i, l_j)) \quad (1)$$

HOG Features [2]

Pixel wise HOG features are computed by applying filters [-1 0 1] and its transpose to a pixel (x, y).Let $\theta(x, y)$ and $r(x, y)$ be the orientation and magnitude gradient obtained. The orientation gradient is then discretized into p bins of the histogram as [4]
$B(x, y) = \lceil (\frac{p \times \theta(x,y)}{2\pi}) mod\ p \rceil$
Pixel wise feature is a p dimensional vector which is obtained as
$\forall b \in 0, 1....p - 1$
$F(x, y) = r(x, y)$ if $b = B(x, y)$
$= 0$ otherwise.
$p = 9$ gives good results [4]
The pixel level feature maps are aggregrated to cells to reduce the size of the feature map. The cell level features thus obtained are invariant to change in bias. This means that the feature does not contain enough information to distinguish whether the object is on which side of the edge.



A)Left side is object B)Right side is object

For this reason the feature vectors are subjected to 4 normalizations with the feature vectors in the neighbourhood. The resulting 36 dimensional vector is subjected to Principal Component Analysis for dimensionality reduction. The 11 dimensional feature vector obtained after the dimensionality reduction contains almost all the information. These feature vectors form the feature map M of the image.

Matching and Score

A filter is an array of weight vectors.Learning the model is essentially learning root filters and part filters from the training set.
The score of the filter F at (x, y) in M is defined as [4]:

$$Score(x, y) = \sum_{(x^{'}, y^{'}) \in F} F[x^{'}, y^{'}].M[x + x^{'}, y + y^{'}]$$

Implementation

The implementation is primarily in MATLAB with some portion in C++. The PASCAL VOC [3] 2006/2007/2009 training sets contains images with bounding box around the objects. The image region inside each bounding box is cropped and resized to a fixed width and height(determined by the

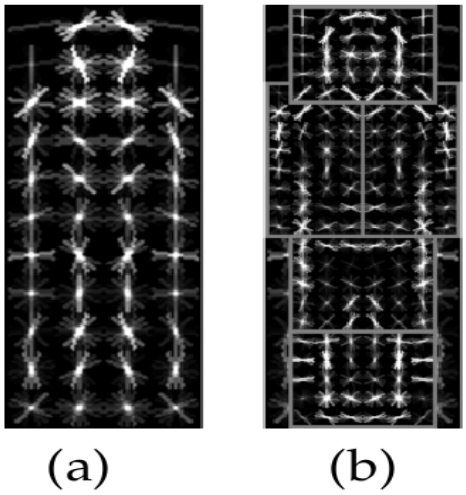
aspect ratio of the bounding box). HOG features are extracted from these resized regions. The obtained features are then clustered to obtain the root filters.
Given a root filter, k $d \times d$ part filters are initialized at twice the spatial resolution in order to capture part details more precisely. By default, $k = 8$ and $d = 6$ which can be modified during training. Individual part locations are selected in two stages:

Greedy Initialization

Part filters obtained above are greedily matched to the image regions in order to maximize the energy map. The energy map [4] is the squared norm of the positive filter weights in each filter cell. The image regions which are matched are not considered later for matching other parts.

Refinement using Stochastic Search

After all the parts are matched, these are displaced one at a time randomly to maximize the amount of energy covered. Note that this displacement costs penalty which is proportional to the magnitude of displacement. When no more energy can be covered, this phase is restarted. This process is repeated many times to avoid selection of local maxima.
An example of model from [4] is shown below:



a)Root filter b)Part filters

Tracker

We implemented tracker by two methods. In the first method, the technique of background subtraction using approximate median is used to keep track of movement in the sequence of images. At any point the background is such that it converges to the approximate median of background in the sequence of images seen so far. Some shortcomings of this method are:
1)It tracks any movement in the entire image and not just the movement of a particular object.
2)It does not work for low contrast images.
3)It cannot distinguish between "actually" moving objects and dynamic objects which are actually stationary.For example in a video with a moving car and a tree(whose leaves are moving due to wind), it tracks both the car and tree leaves.

Due to these shortcomings, we implemented mean shift method [8, 1] for tracking. Objects in motion are characterised by their colour histograms. Intuitively,the object in the next frame will be located somewhere in the vicinity of its location in the current frame. We estimate the histogram of the object in subsequent frames. Mean shift is an iterative procedure which compares the histograms of tracking window in current frame with the histograms of neighbouring regions and maximizes the correlation between them. The weights given to the points are determined by the Kernel function. Bhattacharya distance is used as metric for nearness of two points.

Results

We picked up random google images obtained on searching "people", "humans", "people walking", counted the number of people(occluded and unoccluded seperately) and then counted the number of detections. The results are summarized in this table:

Type	Total Number	Detected	Percentage
All	134	97	72.4
Unoccluded	61	55	90.2
Occluded	73	42	57.6

Figure 1,2,3 show detections by the code we have obtained from [4]. Figure 4 shows background subtraction implemented by us on the Weizmann

dataset. Figure 5 shows mean shift tracker's detections on the Weizmann dataset. Figure 4,5 are frames taken from in-between our result videos.

Some sample detections are shown below

Figure 1 : Bounding box around detected persons

Figure 2 : Detections in images with occlusion

We observed that people far in the background are not detected even if their body parts are not occluded. This is because the colour contrast of these people with the background is low. Edges which define the boundary of body and individual parts become indistinguishable from the background. The detector first extracts the low level features which are essentially these edges. Due to weak low level features, the score of the subsequent matching is not enough to cross the threshhold(the detector algorithm [4] sets a threshhold score and only those detections/matchings are termed successful whose score is above this threshhold).

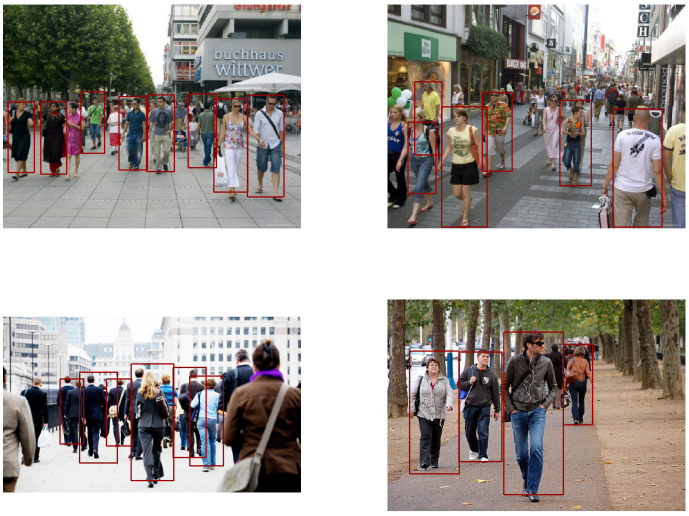


Figure 3 : People far in background undetected

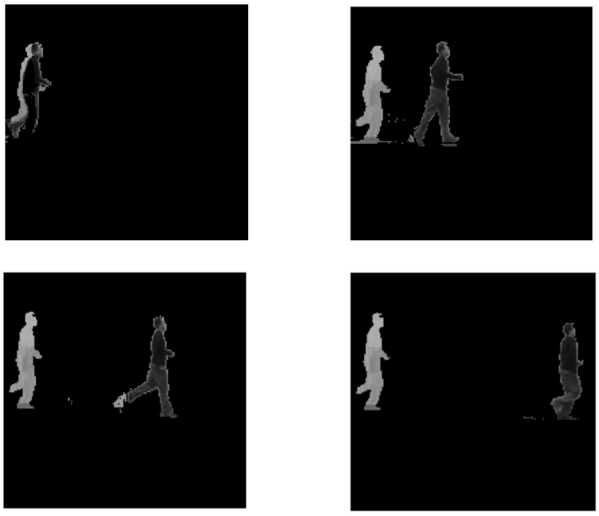


Figure 4 : Tracking using Background Subtraction

Figure 5 : Tracking using Mean Shift Algorithm

Appendix

Using the Code

We obtained the code from project page of [4]. The original code consisted of 73 Matlab and C++ files and the PASCAL VOC Challenge code also had a few Matlab files. The part based structure used by the authors of [4] and the pictorial structures representation in [6] are quite complex structures and took a considerable amount of time to follow along with the code.

Initial Extensions

Initially, we were thinking of improving efficiency of pedestrian detection by training the algorithm on images collected and annotated by us. The images largely consisted of positive instances of pedestrians in various poses(mainly walking/standing on roads). It was expected that with such images as input during training, a precise/robust model for pedestrian detection will be the

result. We wrote a matlab *annotate.m* script which takes as input an image and makes a text file containing annotations in the PASCAL Annotation 1.00 Format. But the code gave runtime errors due to absence of hard negative and false positive instances of annotation which are required by the algorithm during training. So we dropped this idea and extended the detection to Tracking in Videos.

Acknowledgements

We are thankful to P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan for making their code publicly available on the project page [4]. We thank Marcelo Molina ,Adam Kukucka, Zach Clay [1] and Felix Hageloh, Roberto Valenti [8] for their mean shift tracker implementations. We acknowledge valuable suggestions and feedback from Dr. Amitabha Mukerjee, M.S Ram and class.

References

[1] Marcelo Molina Adam Kukucka, Zach Clay. Implementation of mean shift tracker. http://www.hackchina.com/en/cont/17222.

[2] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[4] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/.

[5] Pedro F. Felzenszwalb. Object detection grammars. In *ICCV Workshops*, page 691. IEEE, 2011.

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

[7] Deva Ramanan. Learning to parse images of articulated bodies. In *In Proc. NIPS*, 2006.

[8] Felix Hageloh Roberto Valenti. Implementation and evaluation of the mean shift tracker. http://staff.science.uva.nl/ rvalenti.

[9] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511 – I–518 vol.1, 2001.