SIV Project
Motion Detection and HOG Implementation for Human Identification

Davide Dalla Stella
UniTn
davide.dallastella@studenti.unitn.it

Abstract

In this project a human identification task is solved. In particular, the proposed system uses a motion detection algorithm to identify areas where motion occurs within a scene and applies the HOG method to get the feature vector and exploit to classify, using an SVM, the moving object as human or non-human. The report focuses on the functioning of the HOG method and its implementation in the proposed system, analyzing the various stages of image processing and the configuration of the parameters used. In conclusion, the report highlights the potential of the proposed approach for the detection of humans within a scene, with possible applications in the field of surveillance and monitoring.

1. Introduction

Video frames are taken from an IP camera positioned at a hypothetical entrance or area where video surveillance is desired. These frames are analyzed by the system using computer vision algorithms to detect movement and recognize if there are people present. The use of computer vision algorithms for human detection has become increasingly popular in recent years, as it offers an efficient and accurate approach. Although the task has been accomplished by completing the various necessary steps in this project, particular attention has been paid to the creation of the feature vector through the implementation of the histogram of oriented gradients. This means that library were used for the motion detection and classification parts to facilitate the implementation.

2. Problem Statement

The project was implemented using the Python programming language and the OpenCV computer vision library. The frames are captured through an ip-camera and they are processed in a way to enhance the motion detection, simplifying the human classification's task, done using an SVM. The problem can thus be divided into 3 main parts:

1. Motion Detection
2. HOG Computation
3. Classification

In short, at the beginning a frame from an ip camera is taken. This is used with a previous take to check if there is motion through a background subtraction method. The sections of the frame where the motion is detected are taken and for each of them the Hog vector is computed. This feature array is used to feed an SVM binary classifier to identify human or non-human being.

3. Motion Detection

The motion detection is implemented through a Gaussian Mixture-based Background/Foreground Segmentation algorithm that works well with a static background.
This algorithm selects models data as a weighted sum of Gaussians. The first step with Gaussian mixture model is to extract the frames from a video (in our case the frames are directly taken from an ip-camera). The frames are stacked in an array and a dummy background image is initialized with the frame's size. For each point the intensity value is modelled across all the frames as a mixture of two gaussian and, once modelled, the intensity value at the corresponding location in the dummy background is initialized with the mean of the most weighted cluster. This because the most weighted cluster will be the one coming from the background.
In practice, in order to use this algorithm, a Background Subtractor is created thanks to the OpenCV library. Each time a frame is read the subtractor is applied to it by computing the binary mask of the moving object. If the area of the object is greater than a certain threshold then it is plausible that a human is present in the scene (see Figs. 1 and 2).

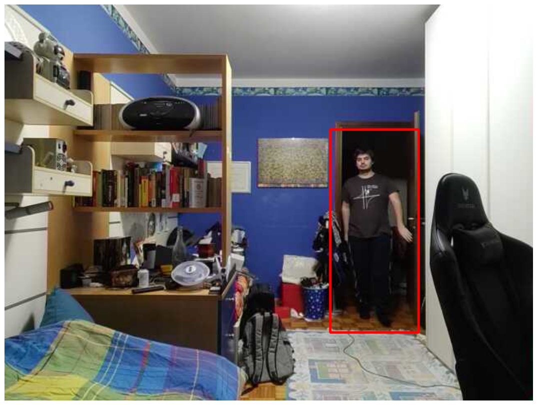

Figure 1. Bounding box around the moving object

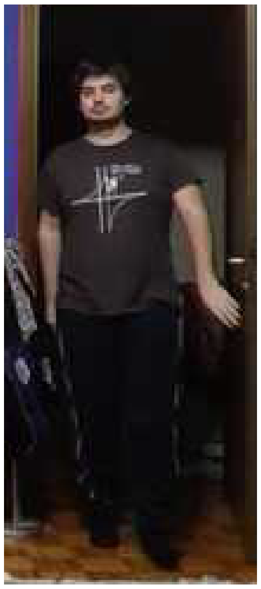

Figure 2. Box where the movement is detected

4. Histogram of Oriented Gradients

The HOG method (Histogram of Oriented Gradients) is a feature extraction algorithm that is often used in computer vision problems such as image classification or object detection. The main idea behind it is that the local appearance and shape of objects in an image can be described by the intensity distribution of gradients or direction of the contours. The implementation of this method is based on the work of Kachouane et al [1].
To compute this vector, several task must be performed: image preprocessing, gradient computation, histogram computation.

4.1. Image preprocessing

First of all, the image must be pre-processed.
The image has been transformed various time in order to facilitate the edge detection. The operations performed are:

0	-1	0
-1	5	-1
0	-1	0

Table 1. Standard Sharp Kernel

1. Resize of the image: done in order to work with objects with the same size and reduce the number of pixels in order to facilitate the hog computation.

2. Convert in grey-scale: done in order to reduce the number of channels for the image and having the pixels values included only in the range [0,255] for just one color.

3. Gamma correction: done to normalize the luminance.

4. Apply a smoothing filter: it reduce the noise in the image.

5. Apply a sharpen filter: done to emphasize the edges of the image and therefore make the gradients in the x and y directions stronger (the kernel used is in table Tab. 1). It is made after the smooth to prevent this filter from over-amplifying the noise in the image.

In Fig. 3 it can be seen how the conversion in grey-scale, the gamma correction,the smooth and the sharpening works.

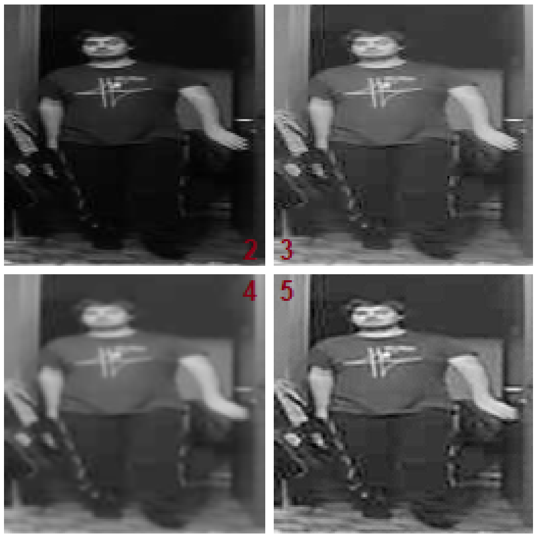

Figure 3. Image transformation

4.2. Gradient Computation

In images, color change is usually associated with the edge of objects. Computing gradients in an image provides information about the intensity and direction of color

-1	0	1
-2	0	2
-1	0	1

Table 2. Sobel y mask

-1	-2	-1
0	0	0
1	2	1

Table 3. Sobel x mask

changes which can then be used to detect and recognize objects within an image. The gradient of the image is calculated by filtering an image with specific kernel, like Prewitt or Sobel. In this work the Sobel filter is applied to the frame. This filter consists of two specific matrices (Tab. 2, Tab. 3) which are convoluted with the original image in order to get the x and y gradients (Fig. 4).

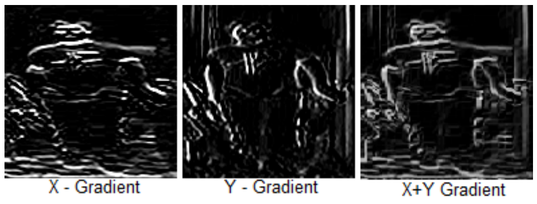

Figure 4. Gradients in X and Y directions

4.3. HOG Computation

For each pixel of the image the magnitude and the orientation are computed by combining the horizontal (G_x) and the vertical (G_y) gradients.

$$G = \sqrt{G_x^2 + G_y^2}$$

$$\theta_G = \arctan(\frac{G_y}{G_x})$$

The image now is divided into small cells and for each one the histogram of gradient orientations is computed. An histogram can be seen as an array where each element corresponds to the frequency for a certain value.
The orientations of the pixels in certain cells are grouped into an 9 orientation bins vector (see Tab. 4) so that each bin represents the histogram of the gradients where the amplitude is given by the sum of the magnitudes weighted according to the pixel orientation.
The weighted values of the magnitude is computed as follow:

$$bin[i] \leq p_o \leq bin[i+1]$$

$$amplitude[i] = \frac{bin[i+1] - p_o}{bin[i]} * p_m$$

0	20	40	60	80	100	120	140	160

Table 4. Orientation bins

$$amplitude[i+1] = \frac{p_o - bin[i]}{bin[i]} * p_m$$

where p_o and p_m are the pixel gradient orientation and magnitude.
An example is provided in Fig. 5
Once the histograms have been computed, they should be normalized using block normalization.
The cells are grouped into 2x2 blocks by stacking their histograms into a single vector that is normalized using L2-normalization:

$$f = \frac{v}{\sqrt{v^2 + \epsilon^2}}$$

where v is the vector and ϵ a constant that make the denominator $\neq 0$.
All the normalized vectors are then concatenated thus obtaining the final hog descriptor.

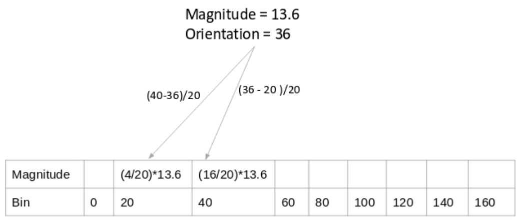

Figure 5. Histogram Computation.
Image taken from [2]

5. Classification

The final HOG-vector is used to feed a SVM Classifier.
A SVM (Support Vector Machine) is a type of supervised learning algorithm used in regression and classification problems. The main objective for it is to find an hyperplane that separates the input data into two classes so that they are maximally distant from the nearest points of both classes. These machines use a kernel function to map data into an higher feature space where a more efficient separation hyperplane can be found, hoping that leads to a better generalization on previously unseen data.
For the SVM a linear kernel is used and it's implementation is made through the sklearn library.
The dataset used to train the SVM is a mixture of two dataset. For the positive samples 360 images are taken frome the Market-1501_Attribute dataset([3]) while for the negative samples 360 frames are taken from the VIRAT 2.0 dataset([4], with frames taken from [5]).

Cell Size	Block Size	Accuracy %
6x6	2x2	82.5
7x7	2x2	83.5
8x8	2x2	87.5
9x9	2x2	86.25
10x10	2x2	86.25
8x8	4x4	90
9x9	4x4	87.5

Table 5. Caption

6. Results

The obtained results are satisfying. To test the performance of the classifier, 40 images for each class were taken from the training dataset. Naturally these images were not used in the training phase in order not to invalidate the results.
For the test the accuracy is computed as:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where $TP+TN$ are all the samples correctly predicted and $TP + TN + FP + FN$ all the samples. Different sizes of cells and blocks were tested to see which one gives better results (Tab. 5).
In the practical test phase, things get more complicated. The accuracy drops and the motivation can be traced back to the boxes returned by the motion detection algorithm which could contain a person as a whole but individually contain only pieces of it.

7. Conclusion

In this project a motion detection + human classification system is implemented.
The motion detection part is implemented through a background subtraction method while the classification part is implemented thanks a linear svm fed with a HOG descriptor.
The main aspect on which this paper focused was the creation of the hog descriptor, highlighting the various techniques adopted for the preprocessing of the frames and how to compute the final vector, also showing how different cell and block sizes can affect performance.

References

[1] M. Kachouane, S. Sahki, M. Lakrouf, and N. Ouadah. Hog based fast human detection. In *2012 24th International Conference on Microelectronics (ICM)*, pages 1–4, 2012. 2

[2] Singh Aishwarya. Feature engineering for images: A valuable introduction to the hog feature descriptor. https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/. 3

[3] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 3

[4] Virat video dataset. https://viratdata.org/. 3

[5] Agi Karasugi. Humanbinaryclassificationsuite. https://github.com/agikarasugi/HumanBinaryClassificationSuite/tree/master/dataset. 3