Pedestrian Detection in Small Device with Fast Hog and Cluster

Kazuki Amakawa

University of Science and Technology Beijing, Mathematics and Physicis Academy

KazukiAmakawa@gmail.com

November 15, 2018

Overview

Pedestrian Detection

Introdution 1

What is Pedestrian Detection?

Pedestrian detection is an essential and significant task in any intelligent video surveillance system, as it provides the fundamental information for semantic understanding of the video footages. It has an obvious extension to automotive applications due to the potential for improving safety systems.

Usage

1) Autonomous vehicles
2) Surveillance camera early-warning
3) Robort vision in Pedestrian Detection
4) Pertreatment of Pedestrian re-identification (re-id)
5) Motion analysis

...

Introdution 2

Object Detection

1) Pre-treatment of image
2) Get the regions may include object
3) Describing these regions
4) Clasify all description
5) Determine the region which include object

In Pedestrian Detection problem, we aim at pedestrain. So the object above is equal to pedestrain. And we have the processing of perdestrain detection

Pedestrian Detection

1) Pre-treatment of image
2) Extracting candidate regions may include human (or just include object)
3) Describing these region(or object)
4) Clasify all description
5) Find pedestrain

Introdution 3

Feature: include edge, texture, color and motion

Human Detection Feature [2]
1) Shape features (pixel level edge-based features)
Disadvantage: Noisy, Pose-specific

2) Region lever edge-based features (Hog)
Disadvantage: Need large calculation, high dimension(difficult for classification)
Pre-processing, PCA, Gabor filter bank(not understant)

3) DNN Feature

Introdution 4

Classification

Processing: TrainData -> Train -> Model <- TestData

|

Classification

Method: SVM(PL-SVM, linearSVM), AdaBoost, DNN(softmax ...)

SVM: Small model and easy control

Pedestrain detection in small device with fish eye camera

Introdution 5

Notice of Pedestrian detection in surveillance camera and early-earning system

1) Continuous images, different from normal image pedestrian detection, surveillance have less moving and it is easy to determine the background and object

2) Realtime processing, in our project, we have to process one image in less than 1 second

3) Bad enviroment: Natural influence: rain, windy; Human influence: too many object; Noisy

Introdution 6

Problem in small device
Less memory and CPU speed, so it is not easy to run a DNN network on these devices

Problem in fish eye camera
1) Distort Image, so other elder model may cannot be used
2) Infrared Model, bad image quality, Noisy

Also, as human structure is different from normal object structure, it have more motion and edge. So it is not easy to build the model.

Introdution 7

Shortage of traditional Pedestrian detection method

High time complex (Hog)

Shortage of DNN Pedestrian detection method

1) Model too large
2) Cannot find C code project
3) Need more train data

Related Work

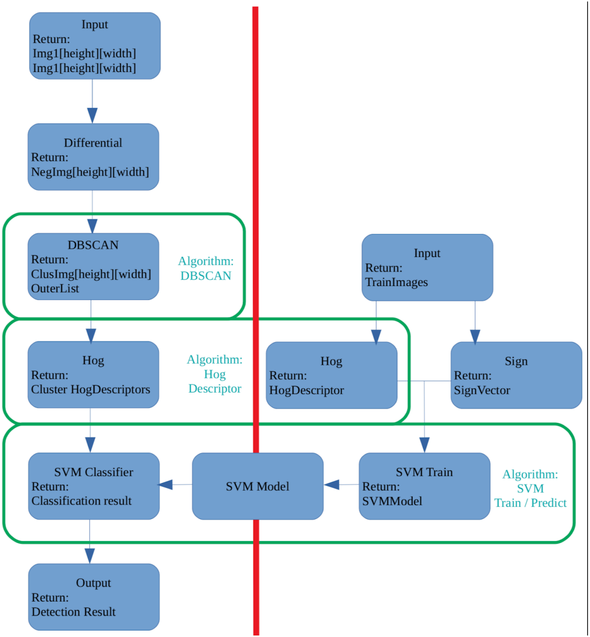


Figure: Fast Hog main processing

Get the object: Pretreatment

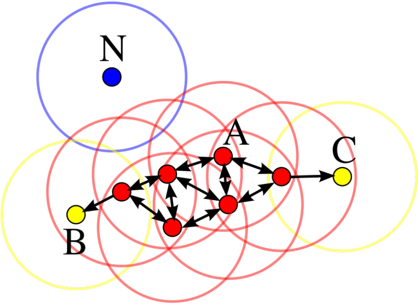
(a) Old Image

(b) New Image

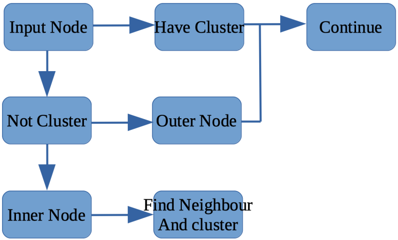
(c) Result

DBSCAN Principle

A point p is a inner point if at least *minPts* points are within distance ϵ.
A point q is directly reachable from p if point q is within distance ϵ from point p and p must be a inner point.
A point q is reachable from p if there is a path $p_1, \ldots, p_n$with $p_1 = p$ and $p_n = q$, where each p_{i+1} is directly reachable from p_i.
All points not reachable from any other point are outliers.
Now if p is a core point, then it forms a cluster together with all points that are reachable from it. Each cluster contains at least one core point; non-core points can be part of a cluster, but they form its "edge", since they cannot be used to reach more points.



(a) Algorithm

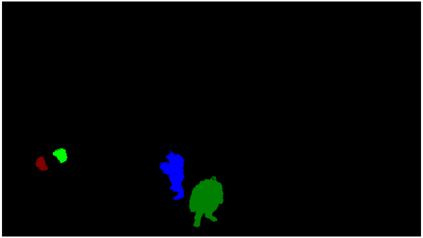


(b) Cluster method

Related Work - Get the object: DBSCAN

Result

(a) Differential Image



(b) Cluster Image

Figure: DBSCAN

Advantage Less Noisy, Fast, Needn't care about k in K-means

Algorithm 1 Hog Descriptor

Input: $Image[Oriheight][Oriwidth]$

STEP1: Down Sample $Image = resize(Image, (height, width))$

STEP2: Gamma transform: $Image[i][j] = GammaTable[Image[i][j]]$

STEP3: Get the *Gradient* and *Angle* image with Sobel operator

STEP4: $BlockImgs = [Block\ New\ Image\ with\ BlockX, BlockY, StrideX, StrideY]$

for Block in BlockImgs **do**

  $Cells = [Get\ the\ Cell\ with\ CellX, CellYineveryBlock]$

  for Cell in CellImgs **do**

    Statistic Histogram of gradient Hisrogram[nbins]

    Normalize the Hisrogram

  end for

  Get the *Hog descriptor in the Block*

end for

Output: *Hog descriptor of the image*

Details of Hog

Gamma Transform

$$Img_{output} = A \cdot Img_{Input}^{\gamma}$$

(Parameter Gamma = 0.5)

Gradient and Angle

$$SobelX = [-1, 0, 1] \quad SobelY = [-1, 0, 1]^T$$

Convolution

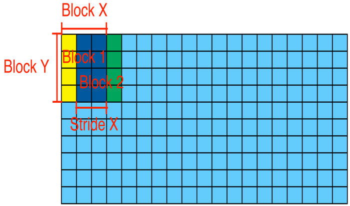
$$GradientX = Image * SobelX$$

$$GradientY = Image * SobelY$$

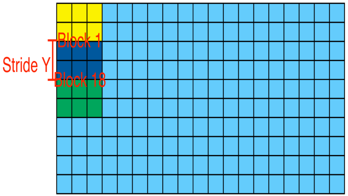
So we have

$$Gradient = \sqrt{GradientX^2 + GradientY^2}$$

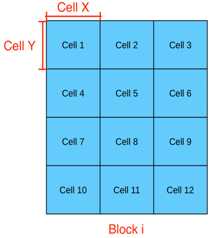
$$Angle = \arctan\left[\frac{GradientY}{GradientX}\right]$$



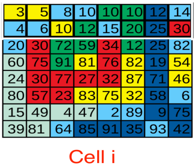
(a) Block X



(b) Block Y



(c) Cells from block



(d) A Cell

Angle

Gradient	3	8	10		30	12	
	5	10	10			25	
	10	14	12				
		4	20				
		6					
		15					
Sum	18	57	42	0	30	37	184
Normal	0.0978	0.3098	0.2283	0	0.163	0.2011	

(e) Statistic Data

Figure: Hog Block and Cells

Related Work - Classification: SVM

Sample Project(C/C++): $https://github.com/cjlin1/libsvm$

Development

Development - DBSCAN

Typically, DBSCAN using the dataset of node. However, as image cluster just have two parameter, x and y. It is not necessary to save all vector which need cluster as a new dataset.

We can list all neighbourhood node as a table in the initial part of the algorithm. And just search the image from the table with DFS(Deep First Search). So the max time complex is $\mathcal{O}(n \cdot m)$, where n is total of node which need cluster and m is the size of table. It is faster than original method $\mathcal{O}(n^2)$

Development - Hog Descriptor

Typically, we need several times Hog network with different parameter to determine the location of person. [1] As we used the DBSCAN for object detection, we don't need the whole image description. We just need the descript for every region(Cluster) as a block. This can also make the model of SVM smaller.

(a) (b) (c) (d) (e)

Figure: Image and Cluster image

Result:

a: $0.268041, 0.028966, 0.259261, \ldots, 0.004802, 0.104433, \mathit{length} = 144$

b: $0.436385, 0.008739, 0.054215, \ldots, 0.021862, 0.255745, \mathit{length} = 144$

c: $0.259077, 0.057242, 0.074549, \ldots, 0.009185, 0.243987, \mathit{length} = 144$

d: $0.349008, 0.100296, 0.149541, \ldots, 0.007501, 0.184265, \mathit{length} = 144$

All Image: $2.26329e^{-11}, 2.595552e^{-11}, 1.03369275e^{-11}, \ldots,$
$2.8235041e^{-11}, 1.8345979e^{-10}, \mathit{length} = 3888$

Infrared Fish Eye Camera Pedestrian Database(IFECPB) and some thoughts

Related Work - About Plan A

As we know, all outer node of DBSCAN cluster must be edge node of every object. So if we using a pixel based detection method, DBSCAN will be a good tool for pretreatment and get the feature.

About Mark

I build a mark system based on DBSCAN to make the mark faster.

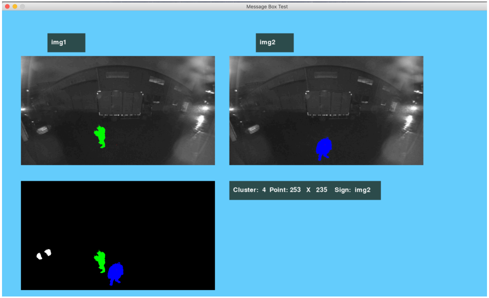


Figure: Image and Cluster image

About Mark

As segmentation work has less relationship with object detection work. [2] It is not easy to connect both of them.
However recently, with the development of calculation, DNN have a rapidly using. And to build the model of DNN, it is necessary to perpare enough data for learning. I think traditional method (segmentation) may be used in this part of work, to make the mark faster and eaiser.

GMM based segmentation in object detection pretreatment and mark.

IFECP Database

As most dataset in pedestrain detection is day time and early-earning system most working in the night, it is not a bad idea to build a dataset include enough image in infrared model.

Characteristic of this data set

1) Include at least 60% night image in infrared model
2) Fishing eye image
3) For pedestrain detection and person re-identification

Result

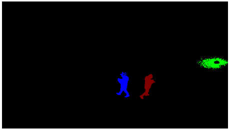
(a) Inp 1

(b) Inp 2

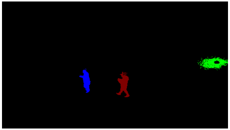
(c) Inp 3

(d) Inp 4

(e) Clus 12



(f) Clus 23



(g) Clus 34

Figure: Image Sequence

Reference

Code: $https://github.com/KazukiAmakawa/FastHog$
*Still under development, not published.

Yuriy Lipetski and Oliver Sidla.
A combined HOG and deep convolution network cascade for pedestrian detection.
Electronic Imaging, 2017(4):11–17, jan 2017.

Duc Thanh Nguyen, Wanqing Li, and Philip O. Ogunbona.
Human detection from images and videos: A survey.
Pattern Recognition, 51:148–175, mar 2016.