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

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

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

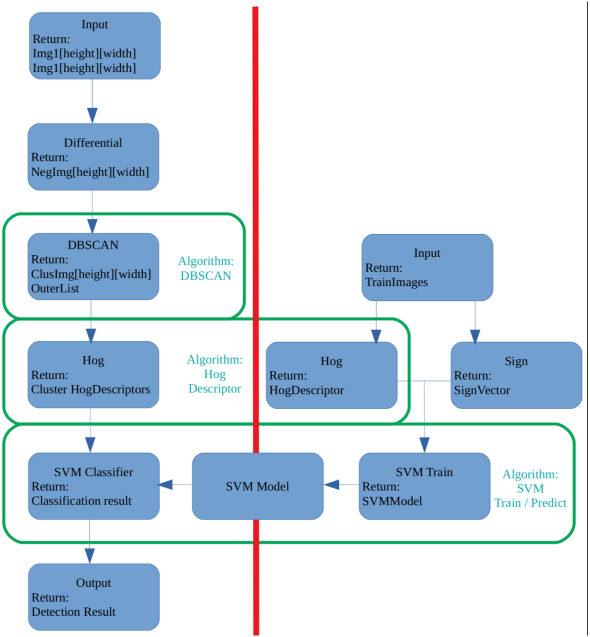


Figure: Fast Hog main processing

Get the object: Pretreatment

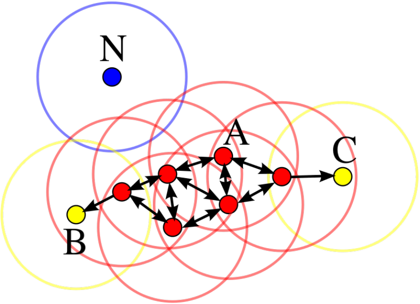
(a) Old Image

(b) New Image

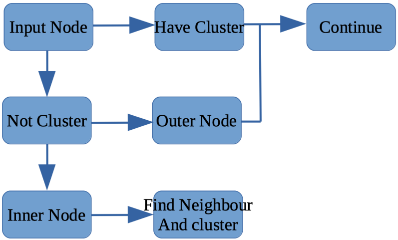
(c) Result

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.



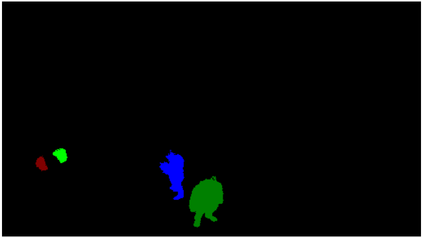
(a) Algorithm



(b) Cluster method

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: $BlockImgs = [Block\ New\ Image\ with\ BlockHeight, BlockWidth, StrideX,$

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

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

Convolution

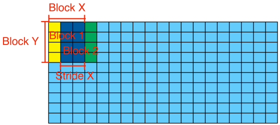
$$GradientX = Image * SobelX$$

$$GradientY = Image * SobelY$$

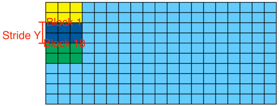
So we have

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

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



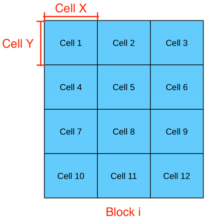
(a) Block X



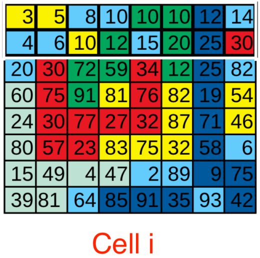
(b) Block Y

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	

(c) Statistic Data



(d) Cells from block



(e) A Cell

Figure: Hog Block and Cells

Related Work - Classification: SVM

Details - DBSCAN

Details - Hog Descriptor

Related Work - About Plan A

IFECP Database

Result

Segmentation in pedestrain detection