Bachelor Thesis

Pedestrian Detection from a Static Camera

Wenqi Hou

April, 2016

Supervisor: Manel Frigola Bourlon

FACULTAT D'INFORMÀTICA DE BARCELONA

Universitat Politècnica de Catalunya
Abstract

Pedestrian detection is a practical and relatively new topic of computer vision. It is widely applied to surveillance, traffic detector and environmental protection. This project focuses on the design and implementation of a pedestrian detection system. The system consists of four modules: movement detector, human and head detectors, position tracker and information extraction. First, the system extracts all the moving objects from the video image. Based on HOG features, we have trained and implemented human classifiers and head classifiers, using SVM and random forest. Also, a tracking module is developed to establish the connection of detections of same people from different frames. At last, the performance of SVM and random forest classifiers is compared in three aspects. They both exhibit their own strength. The datasets used in this project are NICTA for human detection and QUML for head detection.
Table of Contents

Table of Figures ........................................................................................................................................ 6
Table of Equations ..................................................................................................................................... 8

Chapter 1 Introduction .............................................................................................................................. 9
  1.1 Pedestrian detection .......................................................................................................................... 9
  1.2 Difficulties .......................................................................................................................................... 10

Chapter 2 Objective and methodology ................................................................................................... 12
  2.1 Objective ........................................................................................................................................... 12
  2.2 Scope ................................................................................................................................................ 12
  2.3 Methodology ...................................................................................................................................... 13
    2.3.1 Background ................................................................................................................................. 13
    2.3.2 System overview ......................................................................................................................... 14

Chapter 3 State of the Art and Methods .................................................................................................. 17
  3.1 State of the art and related works ..................................................................................................... 17
    3.1.1 Datasets ...................................................................................................................................... 17
    3.1.2 Brief history of pedestrian detection ......................................................................................... 20
    3.1.3 Aspects to improve ..................................................................................................................... 20
  3.2 Feature Descriptor ............................................................................................................................. 21
    3.2.1 Histogram of Gradients .............................................................................................................. 21
    3.2.2 Advantages of the HOG method ............................................................................................. 22
    3.2.3 Implementation of the HOG ...................................................................................................... 23
  3.3 Support Vector Machine ..................................................................................................................... 26
    3.3.1 Introduction to classifiers .......................................................................................................... 26
3.3.2 Introduction to SVM ................................................................. 27
3.3.3 Advantages of the SVM ............................................................ 35

3.4 Random Forest ........................................................................... 35
3.4.1 Introduction to random forest .................................................... 35
3.4.2 Process of random forest .......................................................... 36
3.4.3 Advantages of random forest .................................................... 38

Chapter 4 Design ............................................................................ 39
4.1 General design ............................................................................. 39
4.2 Movement detector ...................................................................... 40
4.3 Detectors ..................................................................................... 40
4.4 Position tracking ......................................................................... 42

Chapter 5 Project Management and Budget ..................................... 45
5.1 Planning ...................................................................................... 45
5.1.1 Project Planning ....................................................................... 45
5.1.2 Estimated time ......................................................................... 46
5.1.3 Resources ................................................................................ 47
5.2 Budget and Sustainability ............................................................ 47
5.2.1 General Considerations ............................................................. 47
5.2.2 Budget ..................................................................................... 48
5.2.3 Sustainability Report ................................................................. 51

Chapter 6 Implementation ............................................................... 54
6.1 Advantages of MATLAB ............................................................. 54
6.2 Implementation .......................................................................... 54

Chapter 7 Results ............................................................................ 58
7.1 Movement detector ...................................................................... 59
7.2 Human detector .......................................................................... 59
7.3 Head detector ........................................................................................................ 60
7.4 Position tracker .................................................................................................... 61
7.5 Comparison of SVM and random forest ............................................................ 66
  7.5.1 Speed .................................................................................................................... 66
  7.5.2 Accuracy .............................................................................................................. 67
  7.5.3 Capacity .............................................................................................................. 70
  7.5.4 Comparison result .............................................................................................. 70

Chapter 8 Conclusion and Future Work ................................................................. 71

References .................................................................................................................. 73
Table of Figures

Fig. 1, System overview .............................................................................................................. 14
Fig. 2, Example of Caltech pedestrian dataset [12] .................................................................. 18
Fig. 3, Example of INRIA dataset [12] ....................................................................................... 18
Fig. 4, Example of ETH dataset .................................................................................................. 18
Fig. 5, Example of TUD-Brussels dataset .................................................................................... 19
Fig. 6, Example of Daimler dataset .............................................................................................. 19
Fig. 7, Example of NICTA pedestrian dataset [1] ...................................................................... 20
Fig. 8, Basic idea of HOG ............................................................................................................ 22
Fig. 9, Process of HOG ............................................................................................................... 23
Fig. 10, Relationship Between Performance and Cell Size, Block Size ..................................... 26
Fig. 11, SVM: Linear situation ..................................................................................................... 27
Fig. 12, SVM: multiple separations ............................................................................................ 28
Fig. 13, SVM: comparison of separation ..................................................................................... 29
Fig. 14, SVM: the planes and the boundary ............................................................................... 29
Fig. 15, SVM: margin width ......................................................................................................... 29
Fig. 16, SVM: non-linear situation .............................................................................................. 31
Fig. 17, SVM: cost variable ......................................................................................................... 32
Fig. 18, SVM: non-linear situation 2 .......................................................................................... 33
Fig. 19, Process of kernel function mapping ............................................................................... 34
Fig. 20, Example process of random forest .................................................................................. 37
Fig. 21, General design ............................................................................................................... 39
Fig. 22, Example of QMUL dataset .............................................................................................. 42
Fig. 23, Path prediction ............................................................................................................... 43
Fig. 24, Gantt chart.................................................................46
Fig. 25, Original frame image .............................................58
Fig. 26, Movement detection .................................................59
Fig. 27, Human detection .......................................................60
Fig. 28, Head detection .........................................................60
Fig. 29, Pedestrian tracking ....................................................64
Fig. 30, Tracking a person in successive frames .......................65
Fig. 31, SVM and random forest: comparison of speed ..............66
Fig. 32, SVM: TPR and TNR .................................................67
Fig. 33, Random forest: TPR and TNR .....................................68
Fig. 34, Comparison of TPR ..................................................69
Fig. 35, Comparison of TNR ..................................................69
<table>
<thead>
<tr>
<th>Equation</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eq. 1</td>
<td>23</td>
</tr>
<tr>
<td>Eq. 2</td>
<td>23</td>
</tr>
<tr>
<td>Eq. 3</td>
<td>24</td>
</tr>
<tr>
<td>Eq. 4</td>
<td>24</td>
</tr>
<tr>
<td>Eq. 5</td>
<td>28</td>
</tr>
<tr>
<td>Eq. 6</td>
<td>30</td>
</tr>
<tr>
<td>Eq. 7</td>
<td>30</td>
</tr>
<tr>
<td>Eq. 8</td>
<td>30</td>
</tr>
<tr>
<td>Eq. 9</td>
<td>30</td>
</tr>
<tr>
<td>Eq. 10</td>
<td>30</td>
</tr>
<tr>
<td>Eq. 11</td>
<td>32</td>
</tr>
<tr>
<td>Eq. 12</td>
<td>33</td>
</tr>
<tr>
<td>Eq. 13</td>
<td>33</td>
</tr>
<tr>
<td>Eq. 14</td>
<td>34</td>
</tr>
<tr>
<td>Eq. 15</td>
<td>35</td>
</tr>
<tr>
<td>Eq. 16</td>
<td>35</td>
</tr>
<tr>
<td>Eq. 17</td>
<td>67</td>
</tr>
<tr>
<td>Eq. 18</td>
<td>67</td>
</tr>
</tbody>
</table>
Chapter 1 Introduction

1.1 Pedestrian detection

Computer vision is the science discipline that deals with perceiving, analyzing and processing visual signals from images or videos, using a computational device. Computers outperform human beings in terms of the ability to calculate and process. But their ability to understand the real world outside the binary inputs is very limited. How to get these powerful machines to understand the environment is one of the most needed and interesting problem of the modern world. Computer vision can be used to detect and track objects, enhance the quality of images, restore damaged images, and build 3D models. It is an interdisciplinary subject that involves machine vision, image processing. Also, it is related to artificial intelligence, machine learning, mathematics, neurobiology, imaging, physics, signal processing and control robotics.

Tracking pedestrians from a set of static surveillance cameras has always been a practical problem that interests researchers, security agencies and governments. A pedestrian detector can be used to calculate the number of people participating in an event, and give warnings of possible dangerous overcrowding. In November 2012 [16], when more than 10,000 people were celebrating Halloween in an indoor stadium in Madrid, the number of people present was much more than the only exit could cover. As a result, there were
three people killed in the stampede. A similar accident happened in Shanghai, China, on new year’s eve, 2015; there were 36 people killed. If there were a pedestrian detection system able to calculate the number of the crowd, the organizers could have been warned and the tragedies could have been avoided. Moreover, the pedestrian detector can be used to control traffic lights when there are too many or too few pedestrians. An automatic pedestrian detector system also can be used to perform statistical studies of crowd behaviors and tracking in an area so as to protect environment.

Pedestrian detection is a benchmark for a variety of detection problems in computer vision. However, this is still an open problem for us to solve, as there is no perfect solution yet. It is still a hard topic to accurately distinguish people in certain types of scenarios.

1.2 Difficulties

Occlusion is a concern. When people walk closely in a group, or when two people encounters face-to-face, it is not easy to separately track them.

Another problem here is when we are using multiple cameras, how we recognize that the objects captured by two or more cameras are in fact a same person.

Also, it is difficult to track a person from non-overlapped cameras. When the target goes out of the range of the first camera, we need to identify this same person when the second camera picks him up.
Low resolution and quality of some outdoor cameras is a challenge too. These outdoor cameras may be blurred by rain and dust, and may be influenced by the sunlight.
Chapter 2 Objective and methodology

2.1 Objective

This project will try to develop a pedestrian detector system to analyze static surveillance camera video, using different computer vision methods to detect people and track their movements.

We first need to identify all the moving objects that appear in frames from their background. Then, we need to establish a feature to exclusively characterize each person. Based on that feature, we need to train and implement classifiers to identify humans. Later, we need to link the same people appearing in different frames, and track each individual’s position from frame to frame. At last, we need to extract the information about these pedestrians.

2.2 Scope

As pedestrian detection problem is still an open problem on which researchers from all over the world are working [3, 8, 9, 10, 11], we cannot expect to solve this problem in just few months. So here we only expect to build a framework pedestrian detection system, to finish each module with an achievable solution. As for more refinement, it is not within our scope.
First, we will extract features and train a classifier to identify people. We will use HOG (Histogram of Oriented Gradient) as a feature and use SVM (Support Vector Machine) or Random Forest to train classifiers. However, there will be false positives, but we accept them, as they can be eliminated in later steps or by a better, larger classifier. We only consider people of relatively fixed size. Those who appear on the far end of the frame will not be identified. Also, people who stand still for a long time will not be tracked.

Second, we will use color or shape information of each person to track them from frame to frame, and we will use the prediction of people’s path as a supportive method. But it is possible that the color or shape information of two different people is similar. In that situation, we accept minor misjudge. When one person is blocked from view by another person, we accept temporary misjudge.

Third, we will try to extract position and movement information of people, and also the total number of pedestrians. There may be minor errors about our information. In the future, we can spend more time on the refinement of the classifier, sample data sets

2.3 Methodology

2.3.1 Background

The detection process can be divided into two stages. First, we need to collect the feature of our object. Second, we need to judge whether this object is what
we are looking for. Typically, there are two approaches to carry out the detection. One is to search for a certain part of the target. In pedestrian detection, we can try to locate body parts like arms, legs, or head. The other is to use a template to compute a certain index, and then use a classifier to judge. In this project, we are using the latter.

2.3.2 System overview

In order to keep track of the project progress, we divide the whole project into several modules.

![System overview diagram](image)

**Fig. 1, System overview**

The first module is to find the moving objects. The second module is to use trained classifiers to detect these moving objects are people. The third module is to develop a position tracker to link people that appear on different frames.
The forth module is to eliminate the false detections and to record the the information that we can from the video. The details about these modules are described in following paragraphs.

To distinguish people from the video background, we start by analyzing the sequence of frames from the video. We first develop a motion detector that marks all the pixels in the present frame that have changed compared to the previous frame.

Then we select only the pixels that describe the outline of moving object by differentiating the image. We choose only the top pixels (pixels that have the highest vertical position) from the moving pixels as sample points (candidates I). These sample points each represents a moving object.

In order to find out whether the moving object is a person, we have to first develop a human classifier. We use HOG (Histogram of Oriented Gradient) method to train a classifier from image datasets of people and non-people. We use the trained classifier to judge a moving object is a person (candidates II). In order to achieve better accuracy, we also train and implement a head classifier to double-check the candidates II. The objects that pass both classifiers are labeled candidates III.

Those objects that are considered people (candidates II) should be tracked from frame to frame. So we need a position tracker that can link these candidates from different frames, and identify the same people that appear in several frames. In other words, we should be able to find some features of people that can serve as their ID for us to track. There are many methods that may achieve
this goal. We can use the color of clothes, or color of hair. We can use their relative positions, as the time interval between frames is really short, and the distance that a person can travel within the interval is limited. Following these features, we can identify a person through a sequence of frames, and record his/her movement.

After these steps and eliminating some false positives, we can gather the information we need from the video. We can create a sample dataset containing information about how many people there are, how many people are going in each direction, or how fast they are walking.
Chapter 3 State of the Art and Methods

3.1 State of the art and related works

As explained above, with its various application, pedestrian detection draws worldwide attention. Many institutes and researchers have contributed to the improvement of the solution. Yet, the problem is not entirely solved, as there is still room for better accuracy and efficiency.

3.1.1 Datasets

Before the work on pedestrian detection even begins, the datasets for the training and the testing of detector should be chosen carefully. The datasets differ from each other in size, angle, color, posture, resolution and so on. So if not the suitable dataset is chosen, we may not be able to generate good results.

There are many useful datasets provided by some institutes. Datasets listed below are some of the most widely-used ones [1]

- Caltech Pedestrian Dataset [2]: it contains 50-pixel or taller, un-occluded or partially occluded pedestrians.
Fig. 2, Example of Caltech pedestrian dataset [12]

- INRIA [3]: it is the oldest dataset with relatively few images and it is the only one that is not originated from video. But it provides many different background settings. Currently one of the most popular static pedestrian detection datasets.

Fig. 3, Example of INRIA dataset [12]

- ETH [4]: Urban dataset captured from a stereo rig mounted on a stroller.

Fig. 4, Example of ETH dataset

- TUD-Brussels [5]: Dataset with image pairs recorded in a crowded urban setting with an onboard camera.
• Daimler [6]: It is captured in an urban setting, contains tracking information and a large number of labeled bounding boxes.

• NICTA Pedestrian Dataset [7]: The dataset contains 25551 different pedestrians, and a large number of negative samples.
Nowadays, the Caltech dataset is considered one of the most important benchmark for pedestrian detection. But in this project, we use the NICTA dataset, as it matches the the angle and frame size of our test surveillance video the best.

### 3.1.2 Brief history of pedestrian detection

Viola-Jones detector [8] was used by Viola and Jones in 2003. They used AdaBoost classifier. The HOG [3] (Histogram of Oriented Gradient) was developed by Dalal and Triggs in 2005. The HOG is one of the most efficient feature for pedestrian detection and it is commonly used as a part of combined features. In 2008, DPM (Deformable Part Detectors) [9] was introduced by Felzenswalb et al. The Caltech benchmark [10] was set up in 2009 to compare different detectors. At about the same time, FPPI (per image) evaluation method [11] replaced the flawed FPPW (per window).

In the recent five years, there are about ten different new methods were introduced. Nearly all the methods can be divided into three families [12]: DPM variants, deep networks, and decision forests. All these families of methods can reach the current best performance.

### 3.1.3 Aspects to improve
• Training dataset: if the training dataset is well-suited, the performance of the detector will be better.

• Classifier: it is obvious that a better classifier will increase the accuracy of detection. But different types of classifiers (like Adaboost and Linear SVM) do not produce very different results [12].

• Supportive data: the use of extra data like stereo and flow information [13] can provide better results.

• Context: using the context of pedestrians (like ground plane constraints, variants of auto-context) can provide minor improvement [14] [15].

• Feature: the better understanding and use of features plays a big part in the improvement of the pedestrian detection accuracy. Now, the most commonly used features are the HOG and color [12].

3.2 Feature Descriptor

3.2.1 Histogram of Gradients

Feature descriptor refers to the indicator calculated to characterize the feature of a certain image area. It should be calculated from the target area, and be fed to the classifier to find out whether the object is a pedestrian. Currently, the HOG (Histogram of Oriented Gradients) and its variations are considered the
most popular and efficient feature describer for pedestrian detection. It was first introduced by Dalal and Triggs [3].

As indicated in its name, the HOG calculated the gradients of small cells of image in each direction, so as to form a feature of the whole area. First, we divide the target area into small, connected cells. Then, we calculate the gradient of each pixel within the cells. At last we combine all the histogram we get from these cells.

![Fig. 8, Basic idea of HOG](image)

To avoid the interference of light and shadow, we can contrast-normalize the histograms. To achieve this, we first divide the area by (larger than cells) blocks. Then we accumulate the histogram of the cells within each blocks. We normalize the histogram of these cells. In this way, we can reduce the influence of lighting in each block, so that we can get a more robust result.

### 3.2.2 Advantages of the HOG method

As we have divided the area into small cells, we can avoid the change of shape and lighting. Because these changes happen in a much larger scale, whereas in small cells, they do not have much influence over the results.
As we are using the accumulative information of the gradient, we can tolerate some body movement. As long as the person is more or less standing straight, small movement will not damage the results.

### 3.2.3 Implementation of the HOG

**Fig. 9, Process of HOG**

First, we convert the area of the image to gray-scale image, to form a three-dimension matrix (horizontal position, vertical position, gray scale of the pixel).

**Normalization**

In order to reduce the influence of lighting, we need to normalize the whole area. We use Gamma compression here. We can set gamma at 0.5.

\[
I(x, y) = I(x, y)^{\text{gamma}}
\]

**Eq. 1**

**Gradient computation**

Compute the gradients of each direction at each pixel inside the cells. The gradient of pixel \((x, y)\) can be represented as:

\[
\begin{align*}
G_x(x, y) &= H(x+1, y) - H(x-1, y) \\
G_y(x, y) &= H(x, y+1) - H(x, y-1)
\end{align*}
\]

**Eq. 2**
where $G_x(x,y), G_y(x,y), H(x,y)$ is the horizontal gradient, vertical gradient and the image value at the pixel.

We can get the magnitude and the angle of the gradient as:

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2}$$

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right)$$

Eq. 3

The most common approach is to use the Sobel mask templates to calculate the convolution in x and y direction.

$$\begin{pmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{pmatrix} \quad \begin{pmatrix}
-1 & -2 & -1 \\
0 & 0 & 0 \\
1 & 2 & 1
\end{pmatrix}$$

Eq. 4

Building HOG for each cell

We divide the area into small cells (typically 6 pixels by 6 pixels). We use 9-bin histogram to record the gradient of these 36 points. So, we divide the 360 degrees into 9 bins. We add the weighted vote (the magnitude of the gradient) of each pixel in each cell to the angle-corresponding bin. We then accumulate the votes of the pixels inside a cell to get the 9-bin histogram for the cell.

Contrast-normalization over blocks

In order to reduce the influence of lighting and contrast changes, we need to normalize the magnitude of the gradient.

As explained before, we first divide the area into connected, overlapped, bigger-
than-cells blocks. The HOG of the whole block is formed from all the cells’ histograms. Since the blocks are overlapped, each cells’ histogram is counted more than once.

We normally use rectangular blocks (R-HOG). A R-HOG can be described by the number of cells in each block, the number of pixels in each cell and the number of bins of each cell. For pedestrian detection, if we set the parameter as: 2*2 cell per block, 8*8 pixels per cell, and 9 bins per cell. So the size of the HOG vector of a block is 4*9=36.

**Collecting the HOGs**

This step is to collect the HOGs of all the blocks inside the area, and combine all the HOG vectors into a big HOG for the whole area.
For pedestrian detection, if we use image area of a person is 64*128 and we set a block every 8 pixels, we will have 7 blocks in horizontal direction and 15 block in vertical direction. There are 36 dimension in each block. So the overall dimension for the HOG vector will be 7*15*36=3780.

It is noteworthy that the best performance (lowest miss rate) is reached when there are 6*6 pixels per cell, 3*3 cells per block. But this set of parameter will need a HOG vector of 12312 dimensions, which is too computational demanding. So, we choose the relatively less efficient 3780 HOG vector to avoid too much computation.
3.3 Support Vector Machine

3.3.1 Introduction to classifiers

Classifiers are the machine learning tool that we use to judge whether an object belongs to a certain group of objects. We first feed a training set of data and corresponding output to feed the classifier. The classifier is trained to understand a pattern of the relationship between data and its output (whether a sample belongs to the group or not). Then the classifier is expected to give out correct output for more unknown data. This process is referred to as supervised training.

Based on the data we collect from feature describers like HOG, in order to judge whether a certain object is a pedestrian, we need to implement a classifier. The Support Vector Machine (SVM) was introduced by Vladimir Vapnik originally in 1963. When Dalal and Triggs [3] first introduced the HOG, a linear SVM classifier
was used by them. Since then, the HOG+SVM approach is considered as the classic way to solve pedestrian detection. However, there is no clear distinction between different classifiers in terms of the ability to detect pedestrians [12]. So, in this project, we implement two types of widely used classifiers: SVM and random forest. We compare the effectiveness of these two types of classifiers.

3.3.2 Introduction to SVM

3.3.2.1 Linear classifier

Consider the situation below. If we need to classifier all the black dots or all the white dots, clearly we need a straight line like the one in Fig. 11.

![Fig. 11, SVM: Linear situation](image)

If we define black dots as $y=-1$ and white dots as $y=+1$, and the line we need should be $f(x)=w \cdot x + b$, where $w$ and $x$ are vectors, $x$ is the input vector, $a)$. When the dimension of $x$ is 2, $f(x)$ is a line in the 2-dimension space. When the dimension of $x$ is 3, $f(x)$ is a plane in the 3-dimension space. When the dimension is more than 3, $f(x)$ is a $(n-1)$-dimension hyper-plane in $n$-dimension space.
When we need to tell whether a new dot belongs to the group of black or the group of white, we just need to calculate \( \text{sgn}(f(x)) \). The sgn function is defined as:

\[
\text{sgn}(x) = \begin{cases} 
1, & x > 0 \\
0, & x = 0 \\
-1, & x < 0
\end{cases}
\]

Eq. 5

But, it is possible that there more than one line can separate the dots. So obviously, we need to select the line that is the farthest from the nearest dots. In order words, we aim to get the maximum margin between the groups and the line. In this way, we can most clearly separate the two groups.

For instance, between the following two lines, we should select the second one.
The dots circled in red and blue is called support vectors.

As shown in the following fig. 14, f(x) is also referred to as the Classifier Boundary. The Plus-Plane and Minus Plane is where the support vectors are.
The support vectors should be on the Plus-Plane or the Minus-Plane. So they should be on the lines: \( w \cdot x + b = 1 \) or \( w \cdot x + b = -1 \). We can convert the expression to \( y(w \cdot x + b) = 1 \).

The Margin Width can be calculated as:

\[
M = \frac{2}{\sqrt{w \cdot w}}
\]

*Eq. 6*

We need to make \( M \) here as large as possible. So we need to make \( w \) as small as possible. This requirement is equal to the following expression:

\[
\max \frac{1}{\| w \|} \quad \rightarrow \quad \min \frac{1}{2} \| w \|^2
\]

*Eq. 7*

\( \| w \| \) is the 2-norm of \( w \).

The complete expression should be:

\[
\min \frac{1}{2} \| w \|^2 \quad s.t., \quad y_i (w^T x_i + b) \geq 1, \quad i = 1, \ldots, n
\]

*Eq. 8*

This is a quadratic programming (QP) problem, and it can be solved by Lagrange Multiplier method. We directly give out the objective function:

\[
L(w, b, \alpha) = \frac{1}{2} \| w \|^2 - \sum_{i=1}^{n} \alpha_i (y_i (w^T x_i + b) - 1)
\]

*Eq. 9*

The result is:

\[
\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \quad s.t., \quad \alpha_i \geq 0, \quad i = 1, \ldots, n \quad \sum_{i=1}^{n} \alpha_i y_i = 0
\]

*Eq. 10*
This the equivalent optimization problem that we need to solve.

3.3.2.2 Non-linear classification

Sometimes we cannot use only a straight line to separate the two groups. For instance, if we have to perfectly separate the two groups, we have to use a curve in the following situation.

![Fig. 16, SVM: non-linear situation](image)

But if we can tolerate some ‘errors’ and avoid over-fitting, we can set an index $C$ (cost or box constraint) to reflect the extent to which we can tolerate the error.
In fig. 18, we accept the misplacement of 2 white dots and 1 black dot. We can also add a slack variable. So, we can now add a constraint function to the original expression:

$$\min \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{R} \varepsilon, \text{s.t., } y_i (w^T x_i + b) \geq 1 - \varepsilon, \varepsilon \geq 0$$

Eq. 11

The slack variable represents the distance that we allow for a dot to cross the support vector. If the slack variable is set at 0, there will be no tolerance for mistaken crossings. If the slack variable is set at a large value, then we will accept a lot of crossings. C is the cost variable for mistakes. If C is set at a large value, there will not be many mistakes, and vice versa. So, if we have to cope with overfitting, we can just increase the value of cost variable.

Similarly, the equivalent problem is:
3.3.2.3 Kernel function

The other way to solve non-linear separation problem is using kernel function. In other words, we can convert the original linear space into a higher dimensional space, in which the dots can be separated by a hyper-plane. Consider the situation shown below:

Fig. 18, SVM: non-linear situation 2

Clearly, we cannot use a plane to separate these dots. But when we map the dots by:

\[
Z_1 = x_1^2, Z_2 = x_2^2, Z_3 = x_3
\]

Eq. 13

We can operate in the new three-dimension space, using one plane to separate them. The conversion process can be shown as following:
So we can change a little bit of the previous expressions by:

\[ x_i^T x_j = \kappa(x_i, x_j) \]

*Eq. 14*

Here are some examples of kernel functions. The polynomial kernel:
The Gaussian kernel:

\[ \kappa(x_i, x_j) = (x_i \cdot x_j + 1)^d \]

Eq. 15

\[ \kappa(x_i, x_j) = \exp\left(-\frac{(x_i - x_j)^2}{2\sigma^2}\right) \]

Eq. 16

3.3.3 Advantages of the SVM

1. SVM can solve classification problems very efficiently when the training samples are not too many.
2. It can solve high-dimensional and non-linear situations.
3. It is a convex optimization problem that we can search for a global solution.
4. Also, we can adjust parameters to avoid over-fitting.

However, it is sensitive to data missing and is only suitable to a small scale of samples. There is no universal solution to non-linear situations. We have to try to find a suitable kernel function every time.

3.4 Random Forest

3.4.1 Introduction to random forest

Random forest is first introduced by Leo Breiman [20] and Adele Cutler. It is a type of classifier that consists of a number of Classification and Regression Trees (CART). As indicated by its name, the trees of the forest are randomly selected, and there is no direct link between these decision trees. When a new input is being processed, every tree in the forest will decide independently to which
group the input belongs. Then the group that gets the most votes will be the final decision.

For each tree, the training set is randomly chosen from the main training set with replacement, which means that a training sample from the main training set may be selected for more than one tree, and may be not selected at all. When we train the node of each tree, the features that we use are also randomly selected but without replacement. As suggested by Leo Breiman, the number of selected features for each node could be in the order of the square root of the total number of all the features.

3.4.2 Process of random forest

3.4.2.1 Training process

We set training data set S, testing data set T, and dimension of features F. The number of CART is t. The depth of each tree is d, and the number of features used by each node is f. The terminal condition for a node is set as: the number of samples is at least s, and the information gain is at least m.

For tree number i:

1. Select with replacement a training data set S(i) from the main training set S. S(i) is of the same size as S and is used as the sample set for the root node.

2. If the current node fulfills the terminal condition, we set the current node as a leaf node. The decision of this leaf node is the sample group c(j) that has the most samples left in this node. Then we move on to train other
nodes. The ratio of the number of samples in \( c(j) \) and the whole sample set of this node is recorded as \( p(j) \).

If the current node cannot fulfill the terminal condition, then we randomly select \( f \) dimensions of features from the total \( F \) dimensions. From these \( f \) dimensions, we try to find a feature dimension \( k \) that gives out the best classification performance and its threshold \( h \). For the training samples of the current node, those whose feature \( k \) is less than threshold \( h \) will be assigned to left child node, and the others will be assigned to right child node. Then we move on to train other nodes.

3. Repeat step 1 and step 2 until all the nodes are trained or marked as leaf node.

Repeat step 1 to 3 until all trees are trained. The following fig. 20 is a basic example of a trained random forest.

![Fig. 20, Example process of random forest](image)

### 3.4.2.2 Classifying process
For tree number i, starting from the root node of this tree, when the feature k of the input is less than threshold h of this node, the input moves to the left child node, otherwise, it moves to the right child node. Repeat the process until reaching a leaf node, and the decision is the leaf’s decision.

3.4.3 Advantages of random forest

1. Random forest can handle high dimensional data, without data preparation.
2. During the training, random forest can detect the influence between each features. After the classification, it can identify the most dominant features.
3. We can train random forest through parallel computing. In this way, we can do train it and use it far more efficiently.
4. It is relatively easier to implement.
5. By using random forest, we can avoid some drawbacks of a single decision tree, like over-fitting.
Chapter 4 Design

4.1 General design

In this chapter, we are going to introduce the overall design of our pedestrian detection system. As stated above, the system consists of four modules, movement detector, human and head detectors, position tracker and information extraction.

Fig. 21, General design
The general idea is: analyze the image sequence of the video frame by frame, compare the frames next to each other, and find the moving parts in the difference of these two neighboring frames. Second, classifiers are implemented to judge whether the moving parts we find in module 1 is indeed pedestrians. To improve accuracy, we have used two categories of classifiers in this project: human classifier and then head classifier. Additionally, to compare the effectiveness of SVM and random forest, we use both of these two methods to train our classifiers. Third, we try to establish a link between the same people appearing in several frames. We predict the path of each person, and use color histogram of his/her body to identify each person. At last, we record the information we collect.

4.2 Movement detector

This is a relatively simple step. We simply compare the frames next to each other, and compute the differential image. As we are trying to use both pedestrian and the classifiers in this project, we try to locate the heads. In the differential image, we focus on the pixels that have the highest vertical position, which means there are no other moving pixels above these pixels. We achieve this by eliminating pixels that has other moving pixels above them. After these procedures, we have found the initial candidates (candidates I).

4.3 Detectors

Given the candidates we get from the previous module, we train classifiers to tell which candidates are actually human-beings, and which candidates are
other objects like cars, pigeons or other animals.

We have already explained in chapter 3 that we select NICTA dataset to train our human classifier, as the NICTA dataset best matches our test video. As for head classifier, select the QMUL dataset [18, 19]. It contains head images cropped from the iLIDS pedestrian dataset. It consists of 18667 images, and provides different angles: back, front, left, right, and background. The images are 50*50 pixels, and are not with ideal illumination. But since the HOG features can eliminate the influence of lighting greatly, the QMUL dataset is suitable to our project.

After training the two classifiers, we first feed the human classifier with the 128*64 pixel image area beneath the heads’ locations we get from module 1, to determine the possible locations of human-beings (candidates II). Then, we feed the 50*50 pixel image area beneath the possible locations from the human classifier to the head classifier. At last we can get a relatively accurate position of the heads (candidates III). After the classifying of these two classifiers, we reduce the possibility of false positive.
It is noteworthy that we adjust the thresholds of the classifiers, so that we can get more positive. Because it is not possible that we retrieve false negatives and it is relatively easier to eliminate the false positives.

We use a linear-SVM classifier, but not a SVM with a kernel. Because as stated in [12], using non-linear kernel does not provide any significant improvement over linear ones. We use 200 decision trees in our random forest classifier, as 200 trees are efficient, and enough to reach a decision.

After two types of classifiers, we may still get multiple detection of one same person. So, we calculate the average position of neighboring detections as our final detection position.

4.4 Position tracking
First we predict the path for each possible pedestrian. As the time interval between two frames next to each other is really little (there are 30 frames in 1 second in the test video), we can assume that the velocity and moving direction of the pedestrians to be consistent. Therefore, we can predict the possible path and the current position at time t for each person from the previous two known positions at time t-2 and t-1. Then we search in the vicinity of the predicted position for positive results from the classifiers from module 2. In this way, we can find the corresponding positive results faster.

Then, we get some possible positives that may be the same person appearing in previous frames. So we use the color histogram to characterize each person. We compare the color histogram of the positives and that of that person in previous frames. The positive that has the most similar color histogram as the pedestrian in previous frame is the match that we are looking for. In this way, we can finally be sure which person is the successor to the one in previous frames, and start to link people from frame to frame.
After finding the successors, we can link the detections frame by frame. However, if there is an occlusion or a detection miss, a pedestrian may “disappear” from our sequence of detection, and then re-appear in the following frames. To address this problem, we make a list of all those whose successor cannot be found, and search this list first when we try to find a successor in the next frames. In this way, we can recover the same person that “disappeared” from previous frames.
Chapter 5 Project Management and Budget

5.1 Planning

5.1.1 Project Planning

We have already divided the project into four modules.

1. Movement detector (5 Days=35 Hours)
   To find the moving objects in each frame as candidates.

2. Detectors (15 Days=105 Hours)
   To train a classifier and use it to detect these moving objects (candidates from module 1) are people.

3. Position Tracker (25 Days=175 Hours)
   To develop a position tracker to link people (identified in module 2) that appear on different frames.

4. Information extraction (3 Days=21 Hours)
   To eliminate the false detections and to record the the information we get from the video.

The final step is to evaluate the project’s results and conclude all the information and present them in charts and graphs.
## 5.1.2 Estimated time

<table>
<thead>
<tr>
<th>Stage:</th>
<th>Estimated dedication (days):</th>
<th>Estimated Budget:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning and feasibility</td>
<td>4</td>
<td>5%</td>
</tr>
<tr>
<td>Analysis and design</td>
<td>6</td>
<td>5%</td>
</tr>
<tr>
<td>1 Movement detector</td>
<td>5</td>
<td>10%</td>
</tr>
<tr>
<td>2 Classifiers</td>
<td>15</td>
<td>30%</td>
</tr>
<tr>
<td>3 Position Tracker</td>
<td>25</td>
<td>30%</td>
</tr>
<tr>
<td>4 Extracting information</td>
<td>3</td>
<td>10%</td>
</tr>
<tr>
<td>Final stage</td>
<td>6</td>
<td>10%</td>
</tr>
<tr>
<td>Total</td>
<td>64</td>
<td>100%</td>
</tr>
</tbody>
</table>

*Table 1, Estimated time*  
(Approximately working for 7 hours per day)

*Fig. 24, Gantt chart*
5.1.3 Resources

As the implementation of the project is done on computer, in all stages of the project, I will use the same materials and resources. To better develop the project, I will use the following device and software.

Hardware:
- Apple 13-inch MacBook Pro with Retina Display (Early 2015)

Software:
- Apple OS X 10.11.4 El Capitan
- Microsoft Windows 10 Education
- Microsoft Office 2016
- Microsoft Visio 2015
- Adobe Acrobat Pro DC
- MATLAB

5.2 Budget and Sustainability

5.2.1 General Considerations

In this part, we assess the feasibility of the project, in terms of finance and sustainability. We will assess the costs of human resources, hardware and software resource. Also, we will assess the sustainability of the project in economic, social, and environmental aspects.
5.2.2 Budget

5.2.2.1 Human resources cost

I am going to finish the project by myself, but with guidance of Prof. Manel Frigola Bourlon. So I will work as a project manager, a program developer and a testing engineer. According to the time estimated in the planning part, the estimation of costs is listed below.

<table>
<thead>
<tr>
<th>Role</th>
<th>Estimated list</th>
<th>Estimated days</th>
<th>Estimated hours</th>
<th>Estimated price</th>
<th>Total estimated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Manager</td>
<td></td>
<td>10</td>
<td>70</td>
<td>50€/hour</td>
<td>3,500 €</td>
</tr>
<tr>
<td>Program Developer</td>
<td></td>
<td>36</td>
<td>252</td>
<td>35€/hour</td>
<td>8,820 €</td>
</tr>
<tr>
<td>Testing Engineer</td>
<td></td>
<td>18</td>
<td>126</td>
<td>30€/hour</td>
<td>3,780 €</td>
</tr>
<tr>
<td>Advisor</td>
<td></td>
<td>8</td>
<td>24</td>
<td>60€/hour</td>
<td>1,440 €</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>72</td>
<td>472</td>
<td></td>
<td><strong>17,540 €</strong></td>
</tr>
</tbody>
</table>

Table 2, Human resource costs

5.2.2.2 Hardware cost

We cannot work efficiently and consistently without the help from a set of useful hardware. The hardware will be used to gather data, process data, store data, and display data. The cost of hardware is listed below.

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Units</th>
<th>Useful life</th>
<th>Estimated amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td>MacBook Pro</td>
<td>1,229  €</td>
<td>1</td>
<td>5 years</td>
<td>122.90 €</td>
</tr>
<tr>
<td>Apple iPhone 6</td>
<td>669  €</td>
<td>1</td>
<td>3 years</td>
<td>111.50 €</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>1,898 €</td>
<td></td>
<td></td>
<td><strong>234.40 €</strong></td>
</tr>
</tbody>
</table>

Table 3, Hardware costs
5.2.2.3 Software cost

Also, the hardware can only function correctly with suitable software installed on them. Some software is provided free of charge to students and researchers. However, we still need to include the cost of such software. Because when the project is working practically, the license will need to be purchased.

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Units</th>
<th>Useful life</th>
<th>Estimated amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATLAB</td>
<td>500 €</td>
<td>1</td>
<td>N/A</td>
<td>50 €</td>
</tr>
<tr>
<td>Microsoft Office 365</td>
<td>9.99€/month</td>
<td>1</td>
<td>3 months</td>
<td>29.97 €</td>
</tr>
<tr>
<td>Windows 10 Professional</td>
<td>69.99 €</td>
<td>1</td>
<td>3 years</td>
<td>6.99 €</td>
</tr>
<tr>
<td>Apple OS X 10.11</td>
<td>0 €</td>
<td>1</td>
<td>N/A</td>
<td>Included on MacBook Pro</td>
</tr>
<tr>
<td>Sublime Text</td>
<td>53.45 €</td>
<td>1</td>
<td>4 years</td>
<td>6.68 €</td>
</tr>
<tr>
<td>Adobe Photoshop</td>
<td>24.59€/month</td>
<td>1</td>
<td>3 months</td>
<td>73.77 €</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>623.44 €</td>
<td>1</td>
<td></td>
<td>167.41 €</td>
</tr>
</tbody>
</table>

Table 4, Software costs

5.2.2.4 General expenses

This section includes other costs that we have not include in above sections, like indirect expenses (mainly electricity costs and internet access costs) and other unforeseeable costs.

<table>
<thead>
<tr>
<th>Kind</th>
<th>Estimated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electricity</td>
<td>25 €</td>
</tr>
<tr>
<td>Internet access</td>
<td>50 €</td>
</tr>
<tr>
<td>Unforeseeable costs</td>
<td>200 €</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>275 €</td>
</tr>
</tbody>
</table>

Table 5, General expenses
5.2.2.5 Total budget

The budget including all the costs explained above is listed below. The contingency is set as 2 percent of the present budget. The taxes are calculated at 21 percent.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Estimated cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human resources</td>
<td>17,540.00 €</td>
</tr>
<tr>
<td>Hardware</td>
<td>234.40 €</td>
</tr>
<tr>
<td>Software</td>
<td>167.41 €</td>
</tr>
<tr>
<td>General expenses</td>
<td>275.00 €</td>
</tr>
<tr>
<td>Contingency</td>
<td>364.34 €</td>
</tr>
<tr>
<td>Tax</td>
<td>3,902.04 €</td>
</tr>
<tr>
<td>Total</td>
<td>22,483.19 €</td>
</tr>
</tbody>
</table>

Table 6, Total budget

5.2.2.6 Budget monitoring

To keep track of the project, we update our results and estimations of the budget every two weeks. In other words, we improve our plan with real time results.

The time consumed in different stages of the project may be more than we estimated, so the human resources cost may be more. To address this, we have to work more efficiently in the following stages. Or we can skip some additional features of the project. If we can finish the project ahead of the planned time, we can add some additional features to our project, then the budget may be bigger, but the reward may also be bigger.
There is possibility that the hardware that we are using breaks down during the project. In this case, we will try to make use of the public computers in school’s computer lab.

If the budget goes beyond our estimation, we can start to use cheaper software. For example, we can use free office software like WPS instead of Microsoft Office. We may have to use more software than listed. In that case, we will use more free open source software.

5.2.3 Sustainability Report

In this section, we will discuss the sustainability and viability in economic, social and environmental aspects.

5.2.3.1 Economic impact

The costs of human resources, hardware and software are assessed above. The cost for hardware are relatively fixed. We can work faster and more efficiently. We can skip some additional features, only including some basic information of the pedestrians in the data pool. We use some extra time to add some practical features to the project, so as to produce more effect. We can use cheaper or free software instead of the listed software. But the functions of the cheaper ones are limited.

There are already traffic control and surveillance cameras everywhere. So we do not need to install new cameras. We only need to use the project to analyze the video taken by the cameras. So the costs are relatively low.
5.2.3.2 Social impact

The objective of this project is to keep track of the number of people in a specific area to prevent stampedes. It can be used in various locations like stadiums, train stations, airports, and schools. If working properly, it will eliminate the possibility of loss of life. So, it has a very significant positive influence on people’s life.

In terms of privacy, the camera videos that we use are traffic control cameras or surveillance cameras that are already installed. We will not add more cameras to damage personal privacy. We simply make better use of video data that exists.

5.2.3.3 Environmental impact

One of the applications of this project is that we can record the number of people who enter a natural area. When there are too many people in that area, the environment may be damaged, so the detector can give out a warning. So the project has a positive effect on environment.

The project is mainly relying on programming. So there will not be any harmful materials involved. All the hardware (computer and mobile phone) and software (programming tool, operating system and office software) used in this project can be used in other tasks in the future. The most energy used in this project is electricity for the laptop computer, which is not very energy consuming.
<table>
<thead>
<tr>
<th>Sustainability Planning</th>
<th>Economic Viability</th>
<th>Improved quality of life</th>
<th>Resource analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment</td>
<td>5</td>
<td>8</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 7, Sustainability Matrix
Chapter 6 Implementation

6.1 Advantages of MATLAB

We have programmed this project on MATLAB. MATLAB is a numerical computing environment and programming language. It is widely used in industry. Here, we give some reasons about why we choose MATLAB as our platform.

1. As MATLAB is based on matrix computing, when it comes to matrix and vector computing, MATLAB works more efficiently than using loops.
2. Its computer vision library and tool box are very convenient to use.
3. Its popularity leads to large community where we can find help to some common problems.
4. MATLAB has a great ability to visualize the results. It is a great tool to present our results.
5. On MATLAB, the value of every variable can be displayed and plotted, which is an advantage when we try to debug.

6.2 Implementation

To achieve better efficiency, after developing each module of the project, I combine all the functions into a single script, so as to avoid the time wasted on calling functions. But the script for each module is still listed below.
• PeoDet.m: Load and process the input video, and then track moving objects of module 1. Input: the video. Output: the differential frame sequence, and moving objects (candidates I).

• TestPeo.m: Calculate the HOG vectors of human body and use a human classifier to detect human body. Input: the differential frame sequence from PeoDet.m. Output: the HOG vectors of each candidate and a vector indicating the judgement to each candidate.

• Find2.m: Record the position of all the candidates identified by the human classifier (candidates II) and mark the all the candidates II in the frame image. Input: the frame sequence and the judgement vectors. Output: the position matrix of candidates II, and marked frame images.

• Testhead.m: Similar to TestPeo.m, using the candidates II, to calculate the HOG vectors of heads, and use a head classifier to detect human head. Input: candidates II. Output: a vector indicating the judgement to each candidate.

• Findh2.m: Similar to find2.m, to record the position of all the candidates identified by the head classifier (candidates III) and mark all the candidates III in the frame image. Also, calculate the histogram of colors in order to track people from frame to frame. Input: the frame sequence and the judgement vectors. Output: the position and color histogram matrix of candidates III, and marked frame images.
• PosElim.m: Eliminate multiple detections of a same person, by averaging the position of these detections. Input: detections’ positions. Output: reduced detections.

• All.m: Include the content of all the above scripts and add the position tracking module. Input: the video. Output: the candidates III, linked detections, and marked image sequence.

• Hog_feature_vector.m: Calculate the HOG vectors of the given image matrix. Input: image matrix. Output: corresponding HOG vectors.

• PeoClass.m: Train the human classifier. Input: positive and negative sample human body images. Output: a human classifier.

• HeadClass.m: Train the head classifier. Input: positive and negative sample head images. Output: a head classifier.

• Check.m: Test the human classifiers using 1002 positive sample and 1421 negative samples from the NICTA dataset. Input: samples, and a human classifier. Output: true positive rate, and false negative rate.

• CheckH.m: Test the head classifiers using 1999 positive sample and 1001 negative samples from the QUML dataset. Input: samples, and a head classifier. Output: true positive rate, and false negative rate.

• Fitcsvm.m: Specifically train a SVM classifier. Input: a sample dataset, and corresponding judgement. Output: a SVM classifier.
• TreeBagger.m: Specifically train a random forest (decision tree) classifier. 
  Input: the number trees, a sample dataset, and corresponding judgement. 
  Output: a random forest classifier.

• DrawClass.m: Plot the figure of the performance of SVM and random 
  forest classifiers. Input: detections and time collected from previous 
Chapter 7 Results

In this chapter, we will present the results from each module, and the final result of tracking in image sequence. Additionally, we will compare the performance of SVM classifiers and random forest classifiers. Note that we have crop a little of the upper part of the video frames, as we cannot perform complete detection near the edge. We present the results at 32 seconds of the video.

As shown in Fig. 25, there are many obstacles that may be challenging to your system, like:

- the flying pigeon in the middle
- the human-shaped models in store windows
- the man who only shows half of his body in the bottom-left corner
- one of the most difficult: the lady with two babies in a stroller

So the frames near 32 second will provide a great evaluation of our system.
At last, we will compare the two types of classifiers that we use: SVM and random forest.

### 7.1 Movement detector

![Fig. 26, Movement detection](image)

As shown in Fig. 26, the white dots represent the moving parts. We make sure that only the dots at the top of the corresponding objects (most obviously shown on the pigeon in the middle of the image) are marked. These are the candidates I that we define in previous chapters. Those objects that are not moving are already eliminated in this module, like the human-shape models in store windows. We can also see from the differential image that there are more white dots on that lady with stroller, as there are three people in that small area. As for the man who is blocked by the building, there are only few dots left for him.

### 7.2 Human detector
Fig. 27, Human detection

Fig. 27 is the image result of SVM human detector. As we do not want to miss any detection, we set the parameter so that there will be as many detections as possible. There will be multiple detections corresponding to a same person, but we can reduce that number with head detector and further eliminate them by averaging the multiple detections that are close to each other. From the image, we can see that the pigeon is already clearly excluded. Still, the lady with the stroller on the right side of the image receives many detections.

7.3 Head detector

Fig. 28, Head detection
We can see from Fig. 28 that after the head classifier and averaging multiple detections, we can get a nice, clean detection of most of the pedestrians in this image. However, the lady with the stroller still presents a problem: we cannot detect her two children correctly, given their small size. But we think it is acceptable. After all, this is a very unusual presence. We can solve this problem with a higher-resolution camera.

7.4 Position tracker
The detected pedestrians are labeled from smaller number to bigger number, from left to right, from top to bottom. As shown above, we can achieve a good performance with our system under normal circumstances. The time interval between frames are very small so that we can easily predict the path and possible position of each pedestrian.

The results show that our system can be applied to the tracking of pedestrians in various situation, like surveillance, or traffic control.

However, when it comes to irregular shapes (like the lady with the stroller) or very small objects (like the babies in that stroller), our system cannot correctly detect such objects. This is an aspect that we should improve in the future.
Fig. 30, Tracking a person in successive frames

Fig. 30 shows how we can track each pedestrian in the video by displaying the tracking of pedestrian number 5 in Fig. 29 from 32 second to 33 second of the video. As shown in the images, a person’s change in neighboring frames is limited, and we connect detections by linking the detections in adjacent frames. So we can track the person that we interested in with robust and consistent detection. This proves that our system can be applied to track people of interest.
7.5 Comparison of SVM and random forest

In order to compare these two types of classifiers, we train them with a same dataset from NICTA pedestrian dataset, and compare their performance in speed, capacity and accuracy. The training set consists of 2000 positive samples. To compare the classifiers under different positive sample to negative sample ratio, we change the number of negative samples from 250 to 9000. The testing dataset is also from NICTA dataset. The positive testing set consists of 1422 images. The negative testing set consists of 1000 images.

7.5.1 Speed

In terms of training speed, as shown in Fig. 31, the time random forest classifier need is twice as the time needed by SVM classifiers. In terms of classifying speed, both types of classifiers need a little amount of time. So SVM classifiers are much faster to use than random forest classifiers.

![Graph showing comparison of SVM and random forest training times](image)

*Fig. 31, SVM and random forest: comparison of speed*
7.5.2 Accuracy

To evaluate the accuracy of classifiers, we introduce two terms: the true positive rate (sensitivity) and the true negative rate (specificity).

\[
\text{True Positive Rate} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})}
\]  
\text{Eq. 17}

\[
\text{True Negative Rate} = \frac{\text{True Negative}}{(\text{True Negative} + \text{False Positive})}
\]  
\text{Eq. 18}

In terms of the accuracy of the classifiers, the two classifiers’ performance is comparable.

\text{Fig. 32, SVM: TPR and TNR}
As we can see from the results, both classifiers have a trade-off between true positive rate and true negative rate. So, using a head classifier after using the human classifier is necessary. In this way, we can achieve better performance, eliminating other objects while keeping the pedestrian detections.

Comparing the true positive rate and true negative rate of these two classifiers, we can see from Fig. 34 and Fig. 35 that: while both classifier provide competitive performance, the random forest reacts more to the changing number of samples. When the number of negative training samples are small, random forest detects pedestrians better, and SVM recognizes background better. When the number of negative training samples are large, random forest recognizes background better, and SVM detects pedestrians better. Generally speaking, the SVM is somewhat more consistent.
Fig. 34, Comparison of TPR

Fig. 35, Comparison of TNR
7.5.3 Capacity

As for the capacity of these two classifiers, both classifiers provide enough capacity. Because random forest just adds more nodes to the tree if the training samples are too many or too varied to classify. As for SVM classifiers, we can adjust cost variable to handle more samples.

7.5.4 Comparison result

The two types of classifiers have comparable performances in terms of capacity and accuracy. The training speed of SVM classifiers are significantly better than that of random forest classifiers. But in reality, the classifiers that pedestrian detection systems use are already trained in advance. So, these two types of classifiers are both suitable in practical use.
Chapter 8 Conclusion and Future Work

In this project, we have built a pedestrian system of four modules: movement detector, human and head detectors, position tracker and information extraction. We analyze the video frame by frame. In movement detector module, we have found all the moving objects from the video image. In detectors module, we have trained human and head classifiers, using SVM and random forest. We have implemented them to locate each pedestrian. Both SVM and random forest classifiers display good performance. In position tracker module, we have established the connection of detections of same people from different frames. In information extraction module, we have collected all the information and data we get from previous modules. We have successfully finished each module, and achieved the goal of pedestrian detection. Also, we compare the performance of SVM classifiers and random forest classifiers.

From the result of this project, we can see that the pedestrian detection system can be used to limit the number of people in a certain area, track a person of interest, develop artificial intelligent robots that can walk on street, or analyze the movement pattern of people to improve traffic control. Based on the principles and ideas from our project, there can be a lot of other application of pedestrian detection system.

However, there is still plenty of room for improvement. The problem of occlusion has still not been solved. Our system does not perform well when we
are dealing with unusual situations, like detecting small children. The color histogram that we use to help identify a same person from different frames is not powerful enough. We can implement more sophisticated, accurate features. Also, we can train our classifier better, like using larger datasets.
References


