Real-Time People Counting system using Video Camera

Damien LEFLOCH

Master of Computer Science,
Image and Artificial Intelligence
2007
Supervisors: Jon Y. Hardeberg,
Faouzi Alaya Cheikh, Pierre Gouton

Department of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

UFR Sciences et Techniques
Batiment Mirande
Université de Bourgogne
BP 47870
21078 Cedex
France

P.i.D Solutions
Oscar Nissensgate 6
N-2849 Gjøvik
Norway

Abstract

A people counter is a device used to count the number of pedestrians walking through a door or corridor. Most of the time, this system is used at the entrance of a building so that the total number of visitors can be recorded. People counting system is important for marketing research (pedestrian traffic management, tourists flow estimation) or in security application (in the case of an evacuation, it is essential to know how many people are inside the building at any given time).

Some sensors are used to count people in a building with different advantages and drawbacks and different accuracy such as laser beam, infra-red sensor, thermal sensor and recently video camera.

In the literature overview, it was found that the most reliable and accurate sensor was the video camera but it was also the most expensive and heavy on computing. Several papers have been published this last ten years for counting (or tracking) people by using video processing with different methods (extraction of the pedestrian silhouette, make rectangular patches with a certain behaviour, mounting the camera vertically to avoid occlusion problem, segmentation of human region by using stereo video camera).

The aim of this thesis was to make a prototype of a real-time counting people system video based by using the Matlab-Simulink programming tool. And then try to measure its accuracy and compare to another system based on laser beam sensors. And finally, conclude in which situations this system is more reliable and therefore find its advantages and drawbacks.

Preface

This master thesis was carried out with the Norwegian Colour Research Laboratory in collaboration with the Norwegian company P.I.D Solutions. For security reasons, this fire company asked the Colour Laboratory to build a system which can count how many people are in buildings at any given time. Actually, this company had already a system like this but it could only measure the flow of people – without any distinction about the direction - and its maintenance was extremely expensive.

This big project was divided in 3 parts between four students; Two students, Mathias Savelle and Nicolas Mauduit, worked with the electronics teachers to set-up a people counting system based with laser beam. One student, Yorrick Jansen, was in charge of the software part - implementing a web application and a centralised store database. And finally, I worked on a people counting system based with Video Camera.

During this master thesis, we also had to set-up an operational prototype of the system because of the short delay time; obviously, four months were a too short period to build a full people counting system.

Damien Lefloch, 2007/09/07

Acknowledgements

During this master thesis, I have gotten many good advices. First, I would like to thank my supervisors in particularly Professor Faouzi Alaya Cheick who was a very great help for me, he helps me to begin with the Matlab-Simulink programming tool that I didn't know before and borrow me the different web-cams. Second, I would like to thanks my friend Yorrick for his good advise and good time we have together in Norway. I would also like to thank an another friend of me, Barbora Kominkovà, who helped me and motivated to write this final report. And obviously, thanks to all my family and friends in particularly my father who always believed in me (without you, i couldn't become what I am now).

Presentation of our work

As said before, this project was divided in three big parts. In this section, I will just present you a brief analysis of the full project, its important features and the different existing system.

The first works to do was to talk with the fire engineer about the system he would like to have (what's the main purpose and features), to know why his current system doesn't suit himself. Hopefully, his answers was really clear. He would like a system capable of knowing exactly how many people are in buildings (for example supermarkets) at any given time. The main goal of this is to manage, in the sequel, the different fire exits (add or remove fire exit, get bigger or smaller fire exit depending of the activity of the buildings) so it has to be reliable to be complying with law.

List of features

- The people counting system must fit with the existing Ethernet network of the company.
- A high accuracy.
- Management of several buildings (need a centralised database updated by each building's main computer).
- Evolution (in order to add some extra features like evaluate the impact of an advertisement).
- Reachable from any place (by using a web application connected with the centralised database).
- Easy management of configuration.
- Breakdown management (detect the malfunction of sensors by using the TCP/IP protocol).
- Easy access to statistics for every buildings or every doors of each building with different interval of time (daily, monthly or yearly).
- Use, when it's possible, free technologies to reduce the cost of the system.

Researches of existing people counting system

We didn't know anything about that kind of technologies, then we had to go further in the subject and get some documentation and examples about people counting systems. Here are the main features we found out during our researches.

Besides, theses researches were carried out at the same time we defined the features with the engineer from P.I.D. Solutions. Indeed, we were looking for information while we proposed several solutions and features to the client. Some meetings allowed us to meet his demand with all information we gathered from our survey.

Since our researches were based on the internet, we discovered plenty companies' web sites, among those. Here are the most interesting ones.

Name of the company	***Web site***	***Field***
Acorel	http://www.acorel.com/en/software.asp	People counting systems in public transport, shopping centre
SPSL	http://www.customercounting.com/	People counting, people behaviour analysis
Infodev	http://www.infodev.ca/	People counting in buildings and vehicles
Abtekcontrols	http://www.abtekcontrols.com/	People counting systems with different sensors such as laser beams
Video turnstile	http://www.videoturnstile.com/	People counting systems based on video processing

Table 1*: Interesting companies and their web sites*

As you saw above, several companies set-up people counting systems, besides one of those has integrated a powerful tool to analyse people behaviour at the same time than counting how many people are inside the building. Nevertheless, we are going to focus on people counting people systems which is basically are divided into two parts: sensors (detection and counting) and software (data management).

The different type of sensors

A lot of systems use video cameras to detect people leaving or entering through a door. The video cameras are generally set-up on a wall, between 1,50 and 4 meters above the floor. Since the system get only images from the cameras, an other component required to analyse the images and allow people counting. This device is placed close to each camera in order to avoid sending too much data through the company's network. In fact, take consider of a video camera capable of acquiring a video in RGB24[1] format at 320x240 Pixels[2]. Thus, each image of the video will be coded in 3x320x240 = 230400 Bytes = 225 KB. Now, if this video camera operates in 15 frames per second (FPS), one second of the video will be coded in 15x225 = 3375 KB = 3,3 MB. This is totally inconceivable, in order to send the video data, the process will use 3,3MB/s of the network's bandwidth for one video camera. The network will be saturated. To resolve this problem, the image processing algorithm close to the camera (for example a micro-processor built-in the video camera) and then just send the counting data. But the fact is that this process is complicated and so very costly in terms of computing and money. However, this system is also the most accurate and can be used and improved to other applications like security (detect “dangerous” situations) or marketing (analyse the behaviour of clients).

Infra-red light produced by red LED is another way to count people. A device

1 RGB24 format is composed of three channels (Red, Blue, Green). Each colour of this format is coded in 3 Bytes (one byte per channel) so it can create 2^{24} different colours.

2 A pixel (picture element) is a single point in a images. The screens' resolution is expressed in pixel unit.

composed by several red LED generates infra-red light and another device which includes a sensor detects when a person cross the line made by those two devices. Obviously a micro controller is included with the sensor, and two groups of such devices are needed to deduce the direction of a person. In fact, two lines are needed to deduce whether a person is entering or leaving through a door.

The same principle as the previous one is used for the laser-beam based sensors. The advantage of such systems is a minimum cost but this systems cannot be "intelligent" means that it cannot differentiate humans. And its other drawbacks is the fact if two persons cross the lines in the same time, it cannot identified this two persons. So in order to limit this problem, it is better to place this type of system in "small door" (door that only just one person can cross over it).

The last well-known way to count people is to use thermal sensors. This sensor is able to detect the variation of temperature in its field so it can identified the temperature of humans. The problem is that it is very sensible to small and high environment temperature. In fact, if the temperature of the ambient air is too high then the sensors will become unusable because of too close temperatures (human and air).

Other systems exist like foot step sensor (sensitive carpet or Differential weight systems) or sound sensor but they are not so frequent because they require heavy modification of the environment and significant amounts of maintenance. They also do not provide an easy way of determining directionality of passers-by.

Unmistakably, the most accurate method is the system based with turnstiles (mechanical counters). The accurate is closer of the perfection. But it cannot be used in supermarkets because it creates a barrier to traffic flow and reduce considerably the "feeling of freedom" in a commercial centre (this counting method obstructs people as they entered and exited).

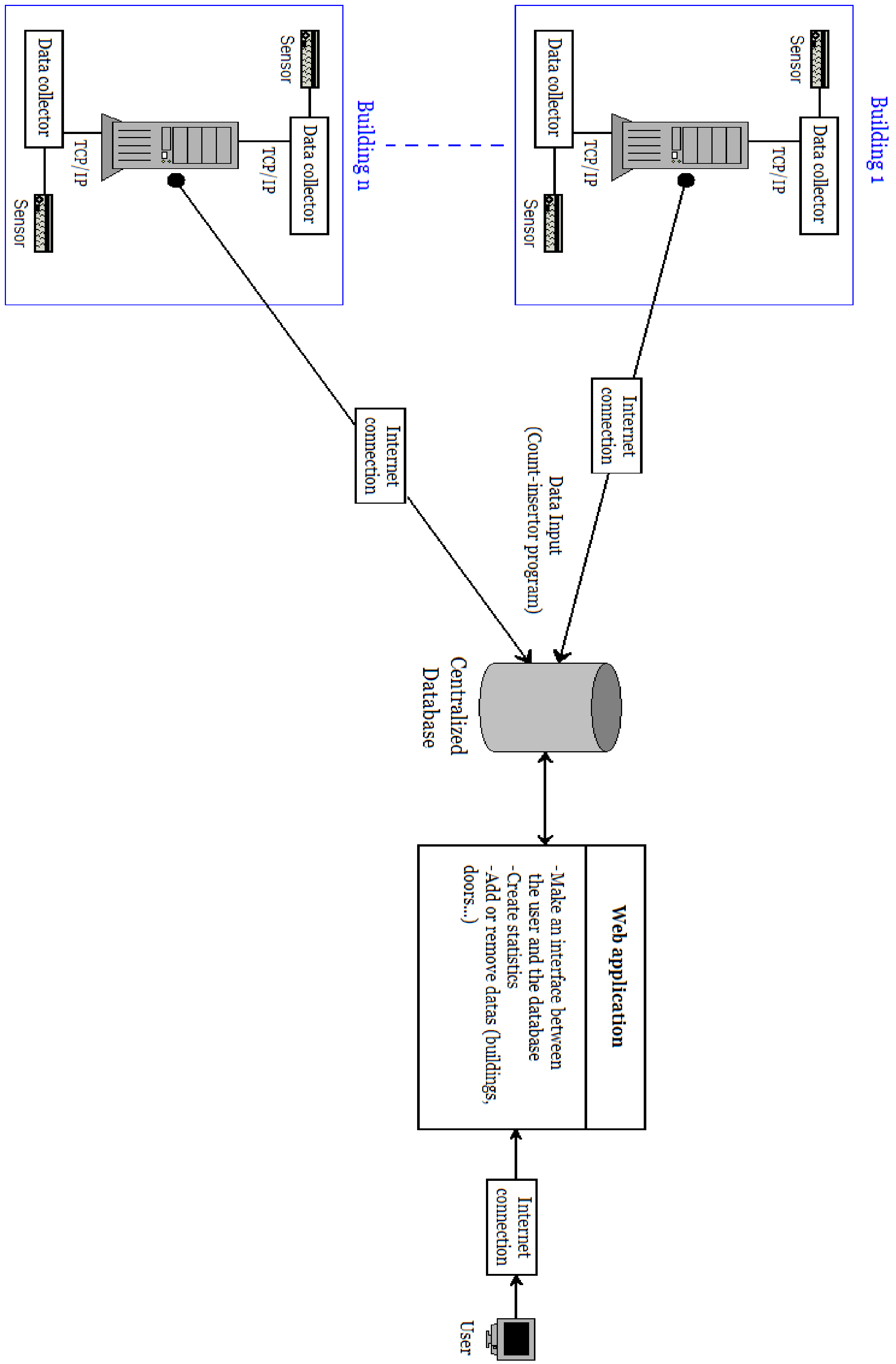


***Illustration 1**: Simple scheme of the entire project*

Table of contents

List of figures

List of tables

1 Introduction

This section covers the introduction and the background for this thesis.

1.1 Topic

The main goal of this thesis was to make different research about existing people counting system based on video camera and build a prototype of this using, in a first time, Matlab-Simulink programming tool. Then, link this system to the centralised database build by Yorrick. And finally discuss and compare it with other existing people counting system based on laser beam build by the two electronics students.

1.2 Introduction

Real-time people flow estimation can be a very useful information for several applications like security or people management such as pedestrian traffic management or tourists flow estimation. The use of video cameras to track and count people increase considerably in the pas few years due to the advancement of image processing algorithms and computers' technology. Several attempts have been made to track people but all those different ways can be classify in three categories of different complexity:

- Methods using region tracking features. To improve this methods some adding a classification scheme of pixels based on colour or textures.
- Methods using 2D appearance of humans (using different models of humans)
- Methods using multiple cameras to make a full 3D modelling.

The third category is more accurate than the two others because it rebuild precisely the scene (so it deals in a better way the occlusion problems) but it is also the most difficult with complex algorithms. Some of the time, this system required a complex camera set-up (calibration) and cannot operate in real-time because the 3D models are too slow. This is why most of the systems used the two other categories.

1.3 Problem description

Today, a lot of research have been published in order to resolve such problem which is count people using video camera. This is not a simple task, there are some situations difficult to solve even with today's computer speeds (the algorithm has to operate in real-time so it makes limits for the complexity of methods for detection and tracking). Maybe one of the most difficult, is people occlusions. When people entering or exiting of the field of view in group, it is very hard to distinguish all the humans in this group.

Thanks to all those research, a lot of companies propose people counting system based on video camera. Their system are very accurate and reliable but are also very expensive. The aim of this entire project is to make a cheap prototype able to count people (obviously, it cannot compete in term of accuracy and performance with the companies' system) and maybe, in a close future, this prototype will become a “truly” people counting system and could be commercialized.

1.3 Thesis structure

This thesis is organized as follows: Section 2 described all the related work done in people counting system based on video camera. In the section 3, the people counting and tracking algorithms are given in details. Section 4 presents the choice of implementation and the advancement of the prototype.

2 Related Work

This part will just give an idea of all the research during this last ten years to resolve this complicated problem which is count people. In fact, thanks to the fast evolution of computing, it is possible to count people using computer-vision even if the process is extremely costly in terms of computing operations and resources.

Computer-vision based people counting offers an alternative to these other methods. The first and common problem of all computer-vision systems is to separate people from a background scene (determine the foreground and the background). Many methods are proposed to resolve this problem. Several suggest counting people systems use multiple cameras (most of the time 2) to help with this process.

Terada et al. creates a system that can determine people direction movement and so count people as they cross a virtual line [TER99]. The advantages of this method is it avoids the problem of occlusion when groups of people pass through the camera's field of view. To determine the direction of people, a space-time image is used.

Like Terada et al, Beymer and Konolige also use stereo-vision in people tracking [BEY99]. Their system uses continuous tracking and detection to handle people occlusion. Template based tracking is able to drop detection of people as they become occluded, eliminating false positives in tracking. Using multiple cameras improve the resolution of occlusion problem. But the problem is the need to have a good calibration of two cameras (when 3D reconstruction is used).

Hashimoto et al. resolve the problem of people counting using a specialized imaging system designed by themselves (using IR sensitive ceramics, mechanical chopping parts and IR-transparent lenses) [HAS97]. The system uses background subtraction to create “thermal” images (place more or less importance depending of the regions of the image; Region Of Interest) that are analysed in a second time. They developed an array based system that could count persons at a rate of 95%. So their system is extremely accurate but with certain conditions. In order to work in good conditions, the system requires a distance of at least 10 cm between passing people to distinguish them and thus to count them as two separate persons. The system also shows some problem in counting with large movements from arms and legs. So this system will be not so appropriate in commercial centre because of the high density traffic when people entering or exiting. In fact, most of the time, person come in supermarkets with their family so make a close group of people which is the most difficult problem to resolve for counting people system.

Tesei et al. use image segmentation and memory to track people and handle occlusions [TES96]. In order to highlight regions of interests (blobs[3]), the system uses background subtraction. It consists to subtract a reference frame (background image previously compute) from the current frame and then threshold it (this algorithm will be more detailed in the *analysis section*). Using features such as blob area, height and width, bounding box area, perimeter, mean gray level, the blobs are tracked from frame to frame. By memorizing all this features over time, the algorithm cans resolve the problem of merging and separating of blobs that occurs from occlusion. In fact, when blobs merge during occlusion a new blob is created with other features but the idea of this algorithm is that in this new blob, it stores the blobs' features which form

3 A blob is defined as a region of connected pixels. The pixels contained in a blob must be locally distinguishable from pixels which are not part of the blob.

it. So when the blobs separate themselves, the algorithm cans assigned their original labels. This system doesn't resolve all the problems but it's a good idea and does not request a lot of computing operations.

Shio and Sklanksy try to improve the background segmentation algorithm (detect people occlusion) by simulating the human vision more particularly the effect of perceptual grouping[4] [SHI91]. First, the algorithm calculates an estimation of the motion from consecutive frames (frames differencing is more detailed in the *analysis section*) and use this data to help the background subtraction algorithm (segment people from the background) and try to determine the boundary between closer persons (when occlusions occurs, make boundaries to separate people by using frame differencing information). This segmentation uses a probabilistic object model which has some information like width, height, direction of motion and a merging/splitting step like this seen before. It was found that using an object model is a good improvement for the segmentation and a possible way to resolve the occlusions problem. But using perceptual grouping is totally ineffective in some situations like, for example, a group of people moving in the same direction at speed almost equals.

Another method of separation of people from a background image is used by Schofield et al. [SCH95]. All the background segmentation algorithm is done by simulating a neural networks[5] and uses a dynamically adjusted spacing algorithm in order to solve occlusions. But because of the reduce speed of neural network, the algorithm only deal with counting people in a specific image. This paper is just an approach of how resolve people counting by using neural networks. Tracking people is not considered.

As simple and effective approach, Sexton et al. use simplified segmentation algorithm [SEX95]. They test their system in a Parisian railway station and get error rate ranging 1% to 20%. Their system uses a background subtraction to isolate people from the background. The background image (reference frame) is constantly updated to improve the segmentation and reduce the effect of lighting or environment modification. The tracking algorithm is simply done by matching the resulting blobs, given by the background subtraction process, with the closest centroids[6]. Means that the tracking operation is operated frame to frame and the label of the blob resulting with the current frame is the same that the blob resulting with the previous frame which has the closest centroid. In order to avoid the majority of occlusions, an overhead video camera is used.

Segen and Pingali concentrate on image processing after segmentation [SEG96]. A standard background algorithm is used to determine the different regions of interest. Then, in each of those areas, the algorithm identifies and tracks features between frames. All the paths of each feature is stored and represent the motion of person during all the process. Then, by using those paths, the algorithm can easily determine how many people crossed a virtual line and the direction of this crossing. This system does not deal with occlusion problems and can be reduce in performance if there is a lot of persons in the field of the video camera. In fact, the paths' data will be big which will complicate the calculation of intersection between the line and all the paths.

Haritaoglu and Flickner adopt an another method to resolve the problem of real-time tracking of people [HAR01]. In order to segment silhouettes from the

4 Perceptual grouping is a psychological phenomena and refers to human visual ability to extract some primitives image features of important regions of the image and group them to make a meaningful structure. The human visual system can detect many classes of patterns.

5 In artificial intelligence, neural networks is modelled like a simplified brain's neural network and composed by a lot of simple elements which are connected with different weights. The most difficult part is to find the best weights for each connections in order to resolve the current problem. There is a lot of applications like speech recognition, face recognition, image analysis... The most impressive thing in neural networks is that some connections between simple elements can exhibit a complex behaviour.

6 Centroid of a 2D shape represents its barycentre.

background, they choose to use a background subtraction based with colour and intensity of pixel values. Those informations will help to classify all the pixels in the image. Three classifications are used : foreground, background and shadow. Then all the pixels classified as foreground make different regions. All these foreground groups are then segmented into individual people by using 2 different motion constraints as temporal and global. In order to track these individuals, the algorithm uses an appearance model based on colour and edge densities.

Gary Conrad and Richard Johnsonbaugh simplify the entire people counting process by using an overhead camera (it permits to greatly reduce the problem of occlusions) [CON94]. To avoid the problem of light modification, they use consecutive frames differencing instead of using background subtraction. To limit computation, their algorithm reduces the working space in a small window of the full scene perpendicular to the flow traffic. At any given time, their algorithm is able to determine the number of people in the window and the direction of travel by using the centre of mass in each little images of the window. With a quick and simple algorithm, they obtained very good results and achieved a 95,6% accuracy rate over 7491 people.

3 Analysis

This chapter gives an overview for the research of this thesis. In this section all the different algorithms used to resolve people counting will be explained.

Here a brief flowchart of the entire people counting algorithm (each module of the flowchart will be detailed in their respective part).

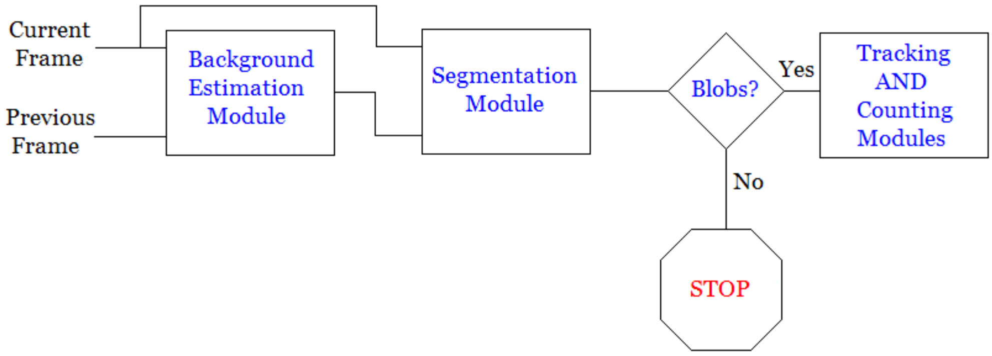


***Illustration 2**: Counting people Module*

3.1 Image Acquisition

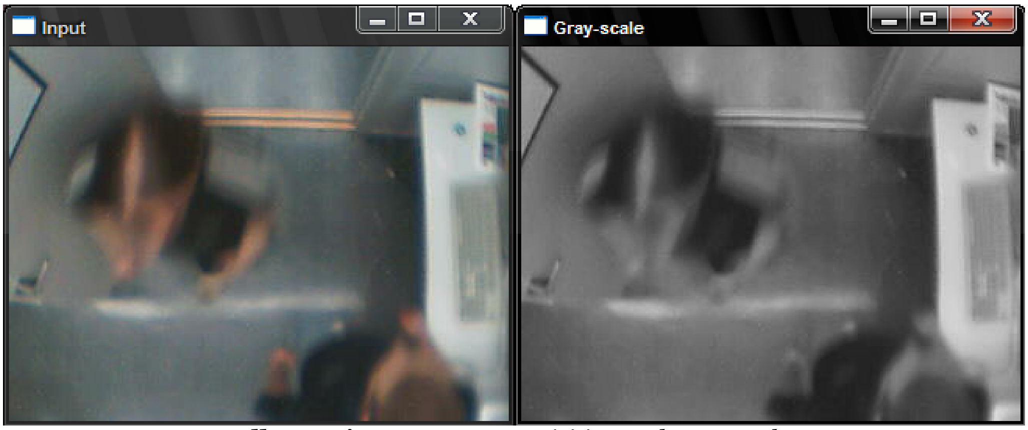


***Illustration 3**: Image Acquisition and Gray-scale*

Image are first required in RGB24 format at 320x240 Pixels. Then the image is pre-processed by transforming it to a gray-scale[7] image. This transformations is easy and just consists to take the average of the three channels of each pixel.

7 A gray-scale image is composed of one channel (intensity). Each intensity (gray colour) is coded in 1 Byte so it can create 2^8=256 different colours.

3.2 Frame Differencing

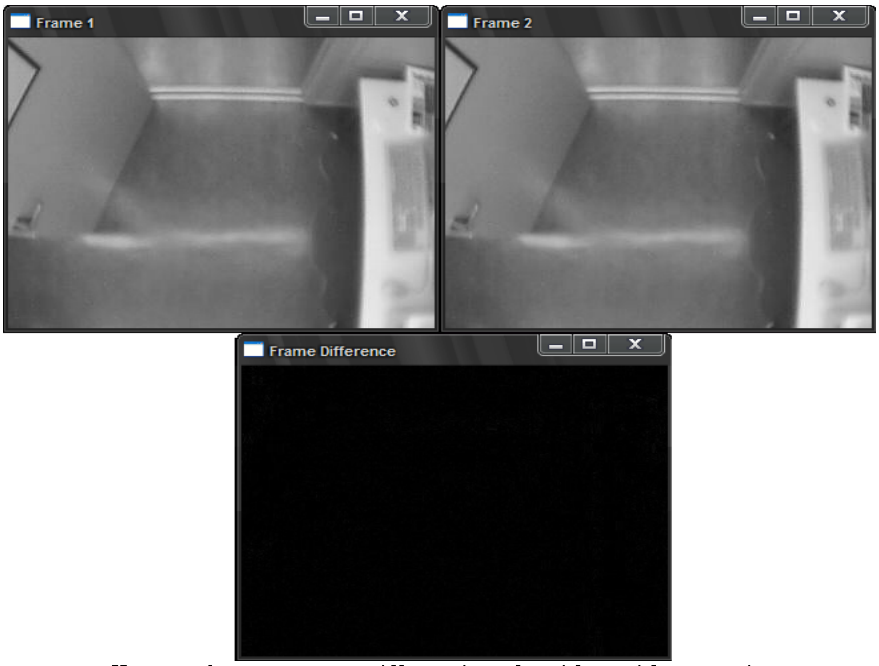


Illustration 4*: Frame Differencing algorithm with no motion*

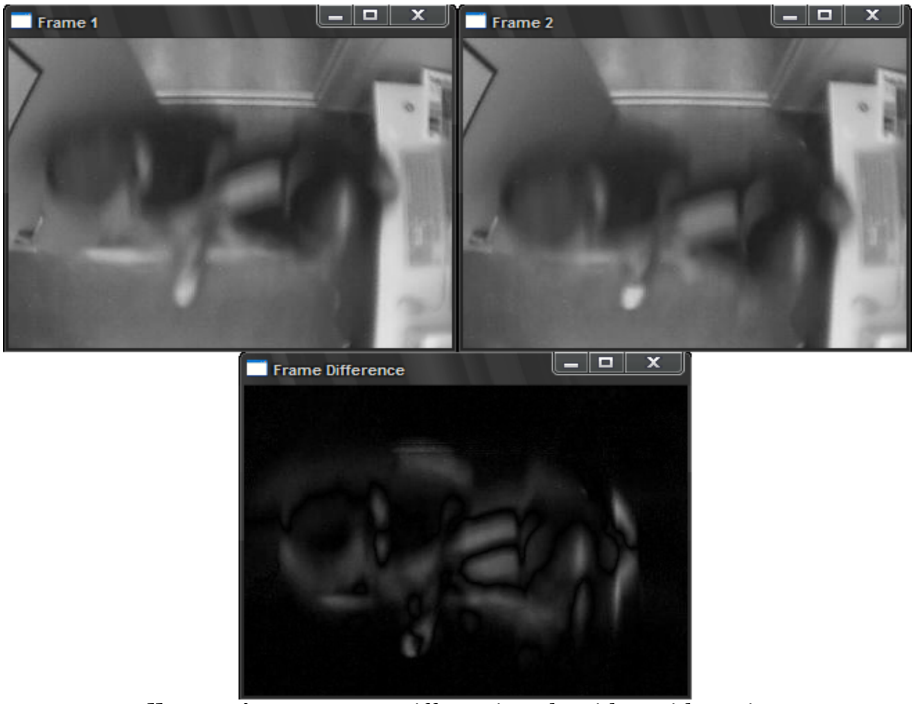


Illustration 5*: Frame Differencing algorithm with motion*

This first step of the people counting algorithm is very important and consists to make a Pixel by Pixel absolute difference[8] between two consecutive frames. The result of this operation is a new image that shows all the differences between this 2 frames. The difference image represents a motion detector. If it's not an empty image (a full black image) then there is modification between the two consecutive frames so there is motion in the field of the video camera. But, due to the quality of the camera used (noises[9]) and some automatic processing, even if no motion is present during the two consecutive frames, the result of the frame differencing process will not be empty. But all the Pixel values resulted by the difference operation will not be big. This means that two consecutive frames are not exactly the same but they are very close. Even if there is noise in the video camera, the “noisy” Pixels have close gray-scale value. A threshold[10] must be introduce to decide of the existence of motion.

Two different ways of applying threshold can be done:

- The first one is a threshold between the difference of each Pixel, all the difference smaller than a certain threshold is considering like a null Pixel (black Pixel). An another to consider that is to take the maximum gray-scale value of the difference image and compare to the threshold. If this maximum value is greater than the threshold then the conclusion is that there is motion in the field of the video camera.

- The second way, is to make a threshold in the full difference image, if the sum of all Pixel's values is smaller than a certain threshold then the difference image is consider like an empty image. The problem of this method is the difficulty to find the threshold because it depends of the resolution of the image (the number of Pixels in the image).

This algorithm can be use also to detect the regions of interest[11] in the image (the regions where there is motion).

8 An absolute difference of two pixels consists to make a difference between the two intensity values of the pixel and take the absolute value. It corresponds at the distance between this two values.

9 Noise is a random, usually unwanted, fluctuation of pixels value in a image due to the quality of the input device (scanner, digital camera...).

10 A threshold is a way to filter informations in two cases (the values which are smaller to this threshold and the values which are greater.

11 Region of interests (ROI) is a way to reduce computing operations by giving more and less importance to the different regions of the image.

3.3 Background Estimation

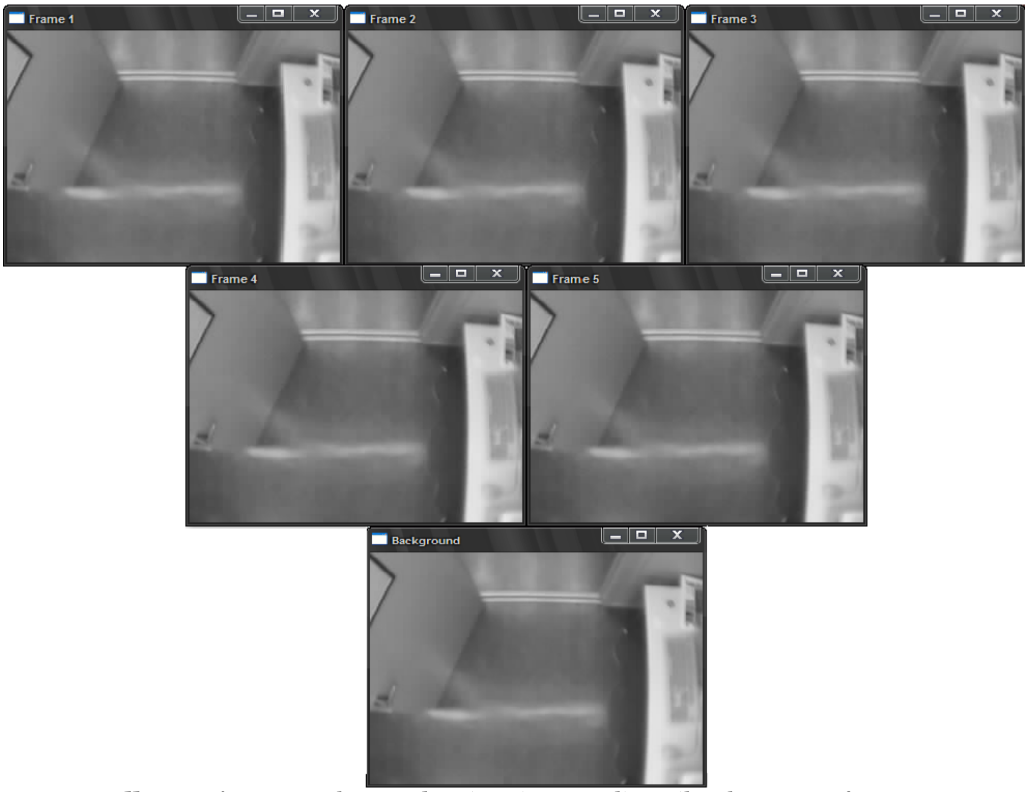


Illustration 6: *Background Estimation - Median Filter between 5 frames*

Before the system attempts to locate person on the scene, it must learn this scene in order to detect motions (big variation of the scene). The background estimation algorithm makes a reference image which represent the background part of the scene view. The background image is fundamental to detect moving objects and tracking them, and will be used to separate the background and the foreground. To avoid the maximum effect of the noises Pixel, a median filter is applying between five consecutive frames in order to obtain a good estimate of the model. For each pixel, the median filter takes their value in the 5 frames, then sort this values and take the median. This process is repeated for each pixel and then will make a new median image (the background image).

The background estimation has to be dynamic, means that the background image must be update. This idea is very important in order to make a good people counting algorithm. For example, if the people counting system is placed in the entrance of one building, some little and slowly modifications occur during all the day and cans parasite the people counting algorithm (more particularly the background difference algorithm). In fact, during the day, the light intensity of the sun changes or ,for some reasons, some objects can be removed or added in the work scene. So, if the background estimation is never recompute, the algorithm will always detect variations.

But before launching the background estimation, the algorithm must be certain that there isn't motion in the field of the video camera during this moment. If there is no motion, the reference image (background image) is updated whereas if there is motion the background image is not updated and will be try to be updated in the next background estimation.

For example, take consider of a gray-scale image format at 3x3 Pixels. Here 5 matrices which represent the 5 consecutive frames:

$$Frame_1=\begin{pmatrix}25 & 53 & 22\\ 2 & 11 & 88\\ 33 & 255 & 65\end{pmatrix}\quad Frame_2=\begin{pmatrix}28 & 55 & 18\\ 5 & 20 & 98\\ 31 & 248 & 70\end{pmatrix}\quad Frame_3=\begin{pmatrix}21 & 50 & 10\\ 0 & 25 & 77\\ 39 & 250 & 45\end{pmatrix}$$

$$Frame_4=\begin{pmatrix}10 & 42 & 38\\ 20 & 30 & 79\\ 45 & 250 & 45\end{pmatrix}\quad Frame_5=\begin{pmatrix}25 & 54 & 22\\ 2 & 11 & 89\\ 33 & 255 & 65\end{pmatrix}$$

By sorting all the values of the 5 frames for each Pixel, 9 vectors are created

$$P_{11}=\begin{pmatrix}10\\ 21\\ \rightarrow 25\\ 25\\ 28\end{pmatrix}\quad P_{12}=\begin{pmatrix}42\\ 50\\ \rightarrow 53\\ 54\\ 55\end{pmatrix}\quad P_{13}=\begin{pmatrix}10\\ 18\\ \rightarrow 22\\ 22\\ 38\end{pmatrix}$$

$$P_{21}=\begin{pmatrix}0\\ 2\\ \rightarrow 2\\ 5\\ 20\end{pmatrix}\quad P_{22}=\begin{pmatrix}11\\ 11\\ \rightarrow 20\\ 25\\ 30\end{pmatrix}\quad P_{23}=\begin{pmatrix}77\\ 79\\ \rightarrow 88\\ 89\\ 98\end{pmatrix}$$

$$P_{31}=\begin{pmatrix}31\\ 33\\ \rightarrow 33\\ 39\\ 45\end{pmatrix}\quad P_{32}=\begin{pmatrix}248\\ 250\\ \rightarrow 250\\ 255\\ 255\end{pmatrix}\quad P_{33}=\begin{pmatrix}45\\ 45\\ \rightarrow 65\\ 65\\ 70\end{pmatrix}$$

The Background image is composed of median values for each vectors P_{ii}

$$Background=\begin{pmatrix}25 & 53 & 22\\ 2 & 20 & 88\\ 33 & 250 & 65\end{pmatrix}$$

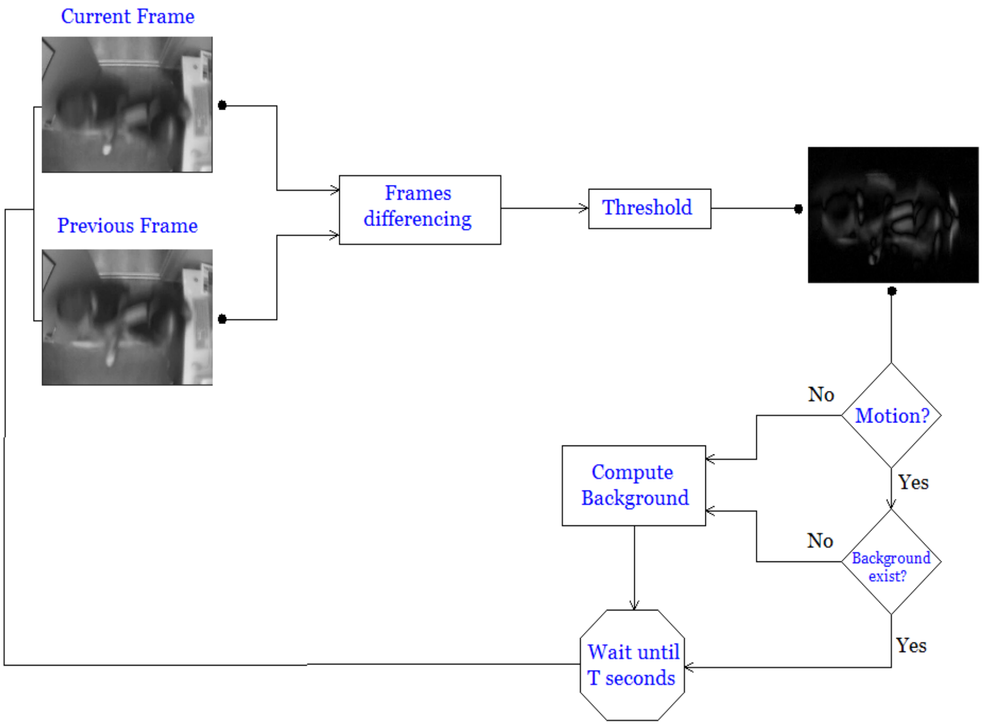


***Illustration 7**: Background Estimation module*

3.4 Background Subtraction and Segmentation

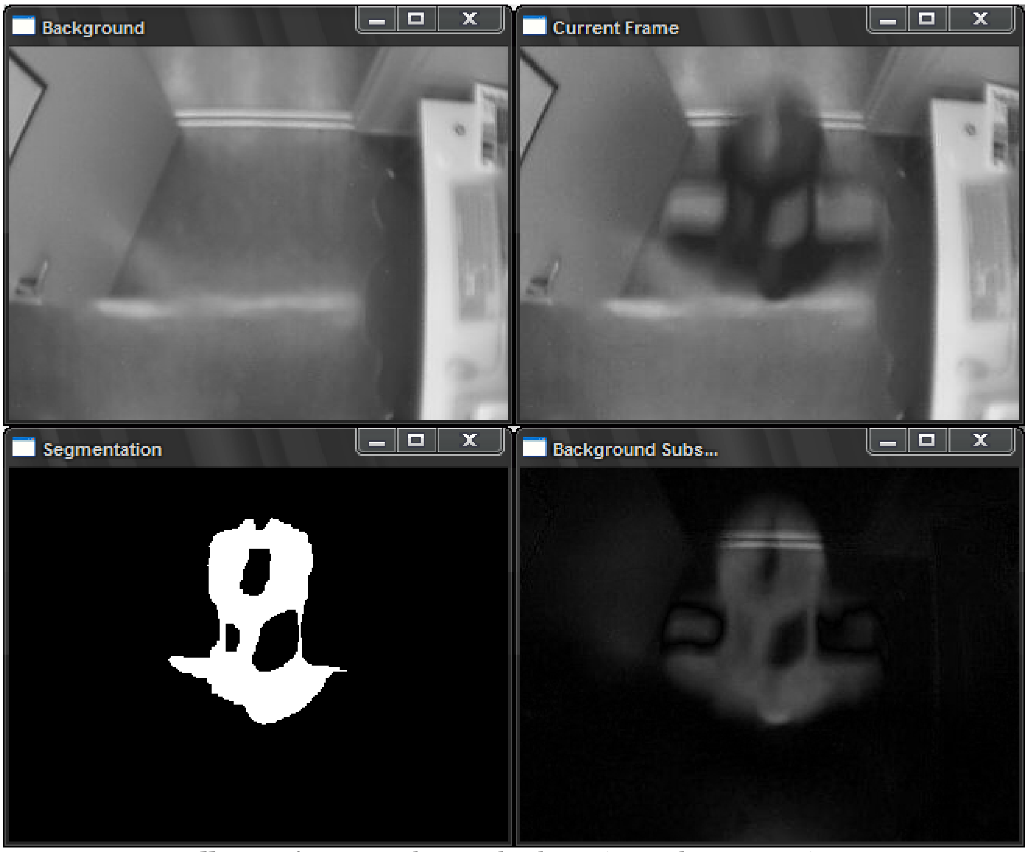


***Illustration 8**: Background Subtraction and Segmentation*

Once the background estimation is completed, the algorithm used to separate the foreground and the background (the foreground just represent the big variation between the current background and the image of the video camera). This process is very close of the frame differencing algorithm but not exactly the same. It is very difficult to separate the foreground and the background of the image by using the frame differencing process (difference between two consecutive frames) cause that it has tendency to just highlight the edges of objects in the foreground so the image analysis will be more difficult afterwards. And an another reason to not use frame differencing algorithm is that it is totally ineffective to detect stationary objects of the foreground. Consequently, people counting system uses the background subtraction step to detect objects of the foreground and frame differencing to detect motion.

The first step of this algorithm, is to make a new image (background subtraction) resulting from the absolute difference between the current background and the current frame. The background subtraction image is a gray-scale image so it has to be transformed in a binary image to make the segmented image (i.e. separation of the foreground and the background). To transform a gray-scale image (255 values) in a binary image (2 values) a threshold must interfere. All the Pixel's values smaller than this threshold is viewed as the background of the scene (value 0). This will eliminate a lot of “noisy” pixels which have, the most of the times, a close value and will eliminate too some of the Pixels which represents the shadows make by the moving objects. In fact, in a gray-scale image, the shadow of an object, most of the time, doesn't change a lot the feature (colour) of the Pixel. So this shadow, in the background subtraction, has a small value.

But this threshold will not eliminate all the “wrong” Pixels (Pixels which are in the background of the scene). The second step of this algorithm, is to make a morphology opening in the binary image (the segmented one). The interest of this morphological operation is that it cans remove small objects created by noise. The morphology opening is composed of two basics operator in the area of mathematical morphology with the same structure element (Erosion and Dilatation). At first sight, make an erosion and then a dilatation with the same structure element cans be taken by an identity. But it's not the case, an erosion is capable of deleting some little objects, so make a dilatation after will not make this objects reappeared so the result of a morphology opening on a binary image is not exactly the same than this original image.

Background subtraction creates images with blobs that correspond to foreground objects.

Now, I will make a small presentation of the different mathematical morphology operations used and illustrate it by some examples.

3.4.1 Erosion

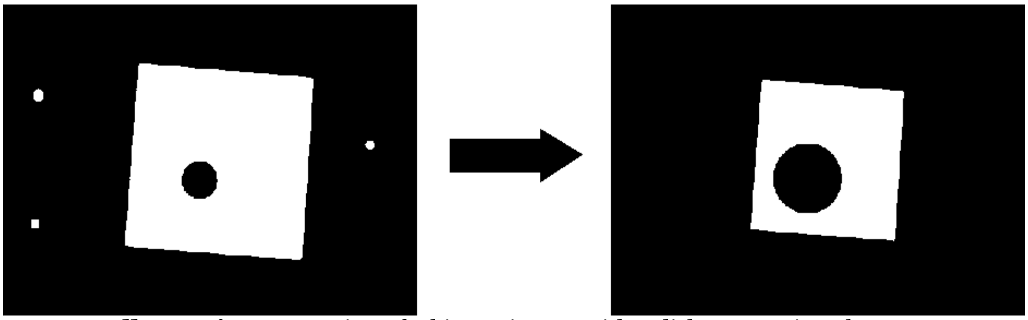


***Illustration 9**: Erosion of a binary image with a disk structuring element*

The first morphological operation used is the erosion. It's a basic operation and its primary feature is to erode away the boundaries of the different foreground regions. Thus this foreground objects will become smaller (little of them will totally be vanished) and holes in objects will be bigger.

Let X be a subset of E and let B denote the structure element. The morphological erosion is defined by:

$$\epsilon_B(X) = X \ominus B = \{x \backslash B_X \subset X\}$$

In outline, all the pixels of the foreground object which can totally contain the structure element B will be contained in the eroded object. For example, take consider of a 3x3 square structure element having its morphological centre the same as the geometrical centre. It is as follows:

1	1	1
1	1	1
1	1	1

Pixels Coordinates

$$\begin{pmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{pmatrix}$$

Illustration 10*: 3x3 square structure element*

To compute a binary erosion, all the Pixels of the foreground must be process. For each pixel of the foreground, the algorithm puts the structure element (the centre of the structure element matches with the pixel) and tests if the structure element is completely contained in the foreground. If it is not, the current pixel will be considered like the background and on the contrary, if it is, the current pixel will be contained in the eroded foreground.

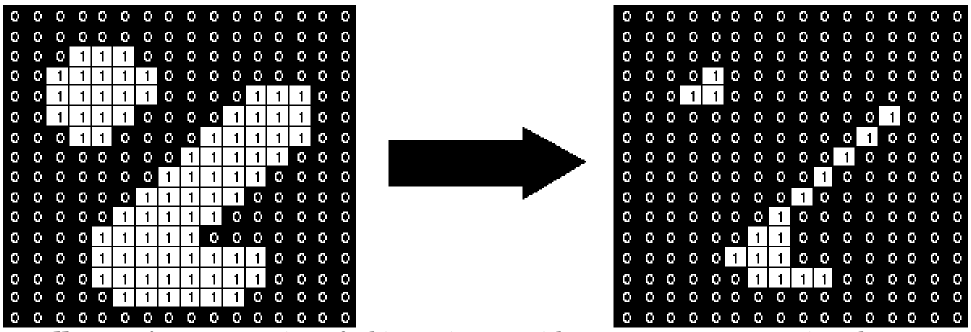


Illustration 11*: Erosion of a binary image with a 3x3 square structuring element*

3.4.2 Dilatation

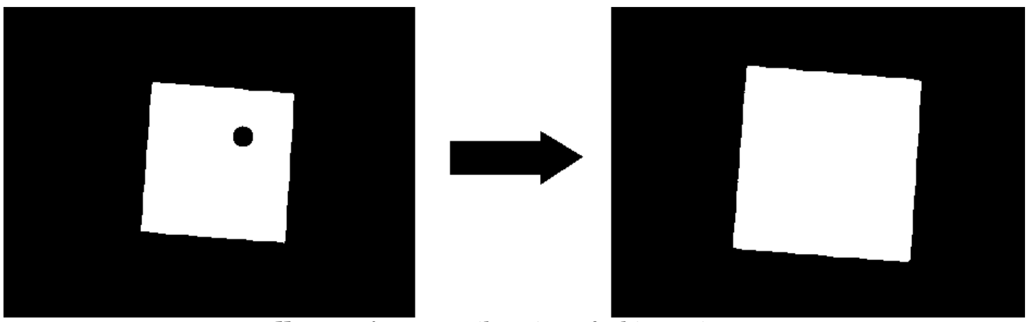


Illustration 12*: Dilatation of a binary image*

Like the erosion, the dilatation is the second basic operation and its primary feature is to dilate the boundaries of the different foreground regions. Thus this foreground objects will become bigger and holes in objects will be smaller (little of them will totally disappear).

Let X be a subset of E and let B denote the structure element. The morphological erosion is defined by:

$$\delta_B(X) = X \oplus B = \{x + b \backslash b \in B, x \in X\}$$

In outline, all the pixels of the background which can touch the foreground regions, by putting on it the structure element B, will be contained in the dilated object.
For example, take consider of a 3x3 square structure element having its morphological centre the same as the geometrical centre (see illustration 7). To compute a binary dilatation, all the Pixels of the background must be process. For each pixel of the background, the algorithm puts the structure element (the centre of the structure element matches with the pixel) and tests if the structure element is in touch with at least one pixel of the foreground. If it is, the current pixel will be considered like the foreground and on the contrary, if it is not, the pixel will stay a background pixel.

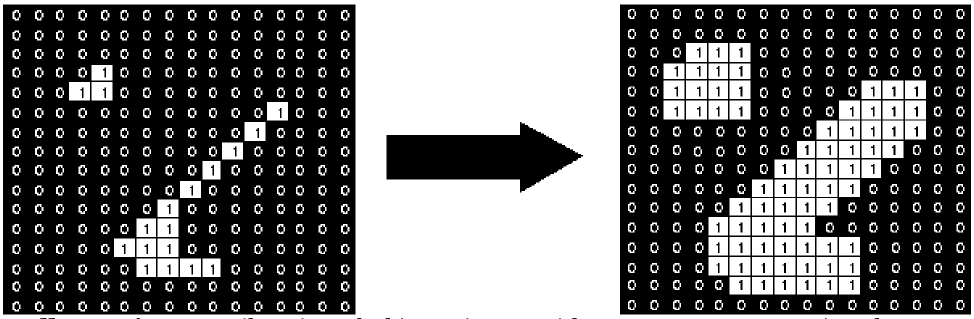


Illustration 13: *Dilatation of a binary image with a 3x3 square structuring element*

3.4.3 Opening

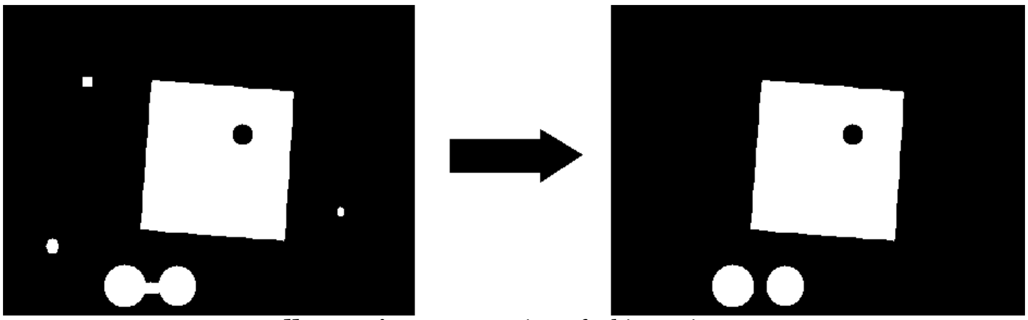


Illustration 14: *Opening of a binary image*

The opening operation is a combination of the two basics operation (Erosion and Dilatation). It's the dilation of the erosion and its primary feature is too eliminate noise (small objects). This operation will separate blobs which are linked with a small layer (see fig.11).

Let X be a subset of E and let B denote the structure element. The morphological erosion is defined by:

$$\Theta_B(X)=\delta_B(\epsilon_B(X))=(X\ominus B)\oplus B$$

In outline, All pixels which can be covered by the structuring element with the structuring element being entirely within the foreground region will be preserved. However, all foreground pixels which cannot be reached by the structuring element without parts of it moving out of the foreground region will be eroded away. As follows, an example of an opening operation with a 3x3 square structuring element.

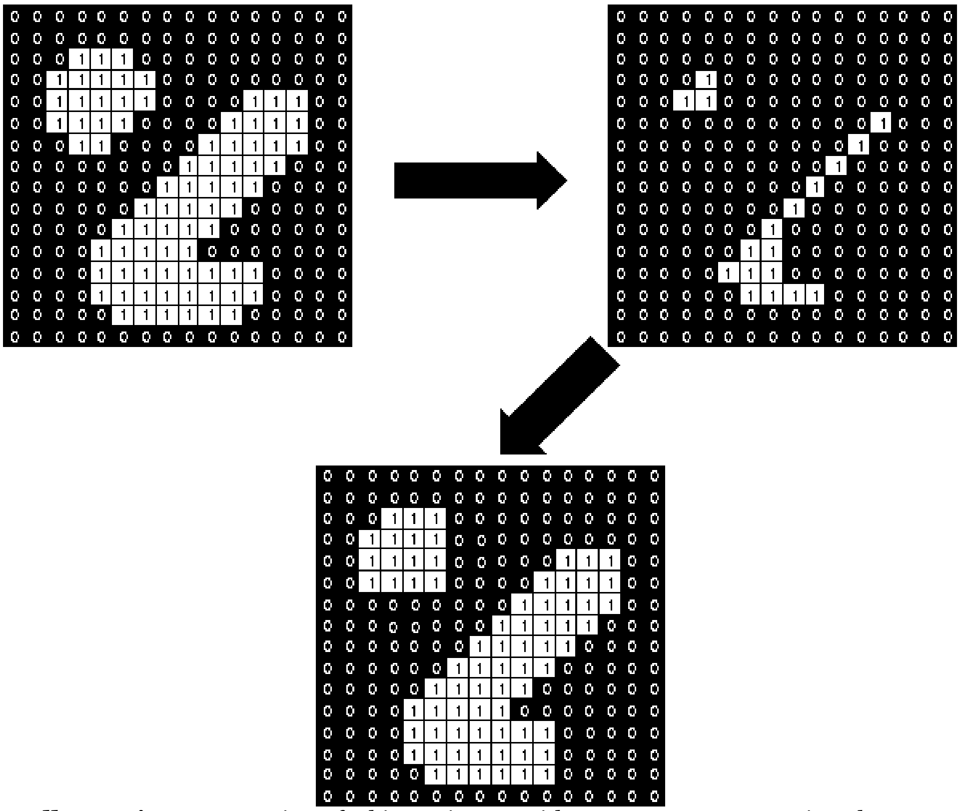


***Illustration 15**: Opening of a binary image with a 3x3 square structuring element*

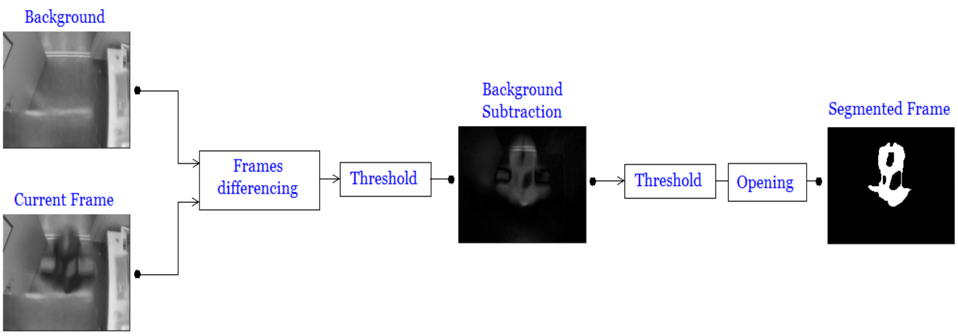


***Illustration 16**: Segmentation Module*

3.5 Tracking

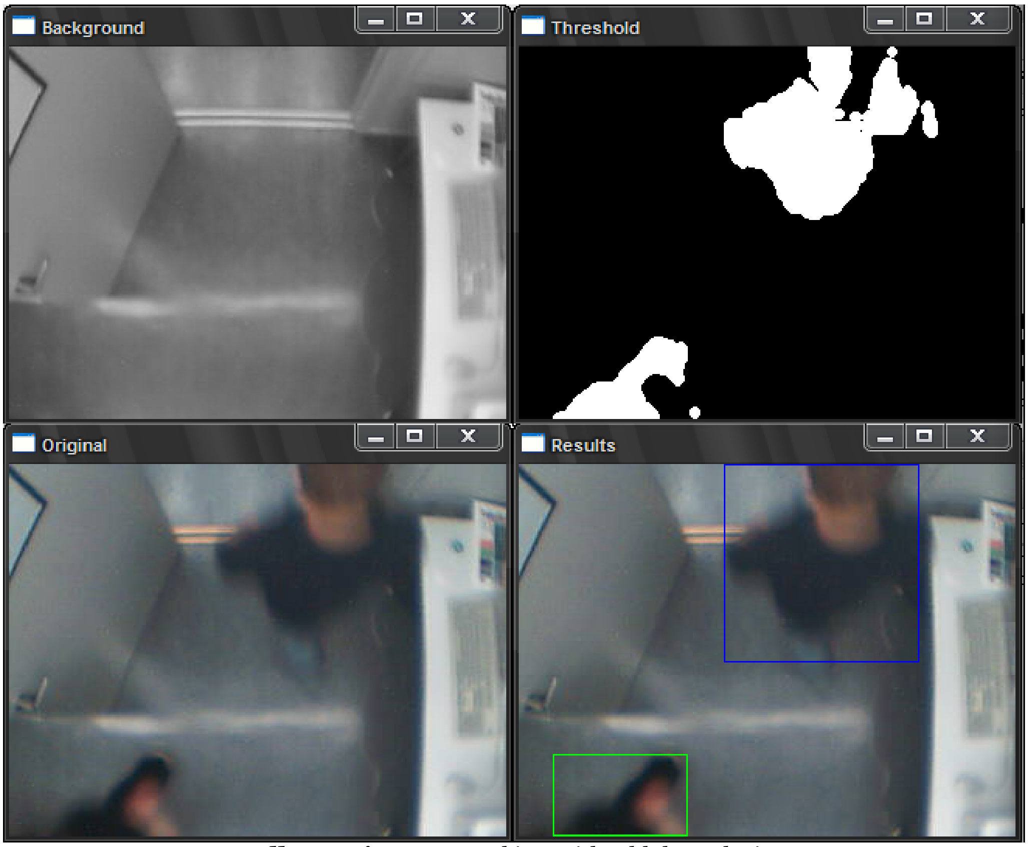
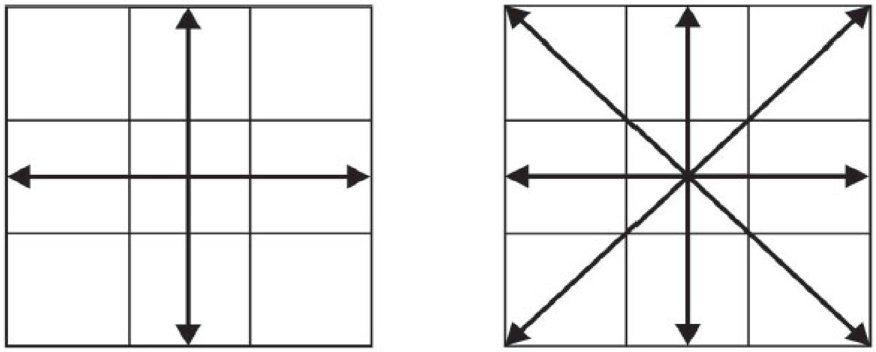


***Illustration 17**: Tracking with a blob analysis*

Once the segmentation is done, an another image processing must be launched in the binary image. In fact, in order to track objects, the first step to do is to identify all the objects on the scene and calculate all their features. This process is called a Blob Analysis[12].

It consists to analyse the binary image, find all the blobs present and compute statistics for each one. Typically, the blobs features usually calculated are area (number of pixels which compose the blob), perimeter, location and blob shape. In this process, it is possible to filter the different blobs by their features. For example, if the searching blobs have to have a minimum area, some blobs can be eliminate with this algorithm if they don't respect this constraint (it permits to limit the number of blobs, thus reduce the computing operations). Two different ways of connection can be defined in the blob analysis algorithm depending of the application. One consists to take the adjacent pixels along the vertical and the horizontal as touching pixels and the other by including diagonally adjacent pixels (*illustration 18*).

12 Blob analysis is the identification and study of the different regions of connected pixels in the image. The algorithm discerns pixels by their value and classifies them in two categories: foreground (typically the non-zero value) and background (pixels with a zero value).



Lattice with Four connections　　Lattice with Eight connections

Illustration 18*: Two rules for touching pixels*

Setting the rules for touching pixels is important because the outcome of the blob analysis can be different. For example (*illustration 19*), the group of pixels would be considered as one blob if the algorithm used the lattice with eight connections and as two different blobs if it used the other lattice.

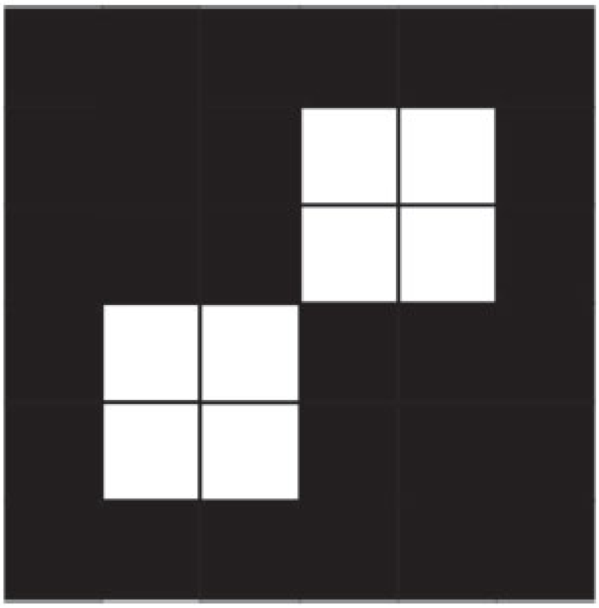


Illustration 19*: Importance of the lattice used*

The performance of the blob analysis algorithm depends totally of the quality of the segmentation. With a bad segmentation, the blob analysis can detect some not interesting blobs or worse can merge some different blobs due to lighting condition or noise in the image.

While all the bounding box[13] are created, the tracking process put a label[14] for each one in order to memorize the different objects present to the scene. But, too keep the

13 A bounding box is a the smallest rectangular shape which surround blobs.
14 A label is an identification (for example a number) given to each objects.

good labels for each blobs in a sequence of frames, the algorithm must memorize some features of each blob. The most simple way to do this is to memorize the position of the centroid, bounding box, width and height for each blob and then in the current frame try to find the blob which is the best matching. But sometimes, it is impossible to find a good matching blob, for example in the case of merging blobs. So when this case is detected, the algorithm has to determine with which blobs this new merging blob was created. Like this, the algorithm does not lose informations about blobs in the scene and this is fundamental for the counting process.

3.6 Counting

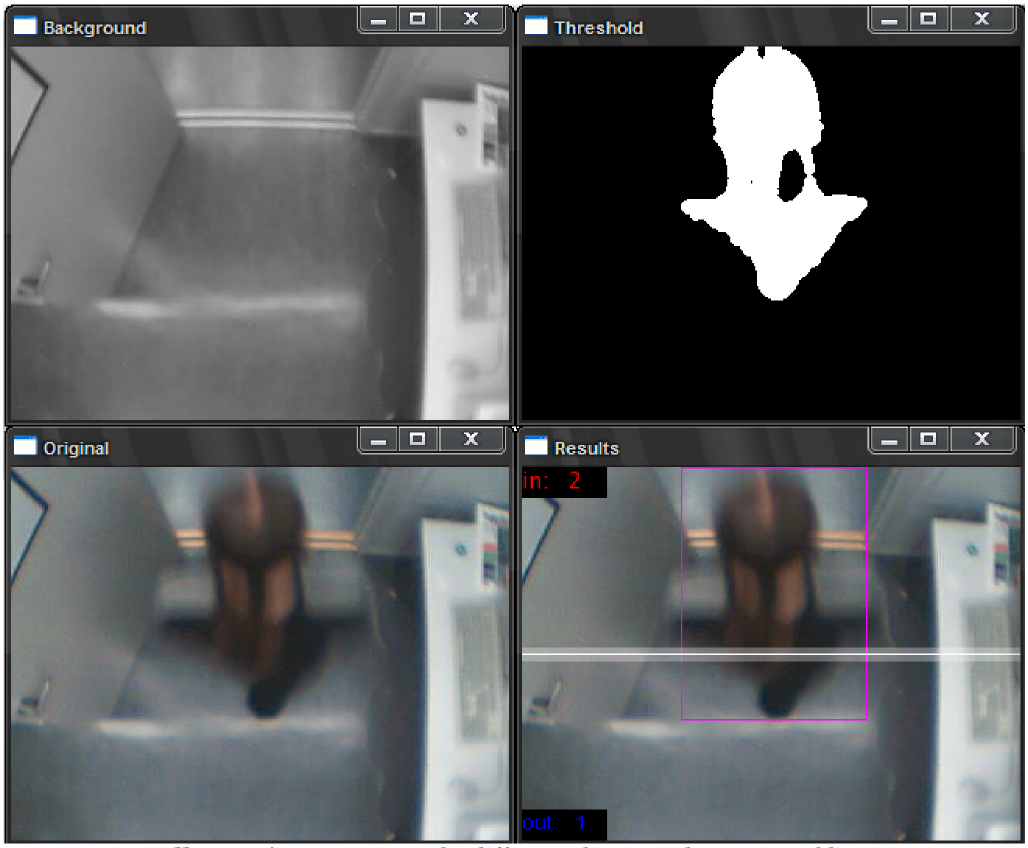


***Illustration 20**: Count the different objects with one virtual line*

In this paragraph, I will just explain the counting process when an overhead video camera is used because my model was made for this situation (the different choices are explained in the *implementation section*). But all the previous algorithms are able to work in any other situations (it is just the counting which depends of these different situations).

The counting process consists to determine the direction of blobs which cross a virtual line in order to increment the good counter. There is two different way to make this process in the case of using an overhead camera.

The first one is to use two virtual lines which represent the Entering (IN) and the Exit (OUT). If a blob cross over completely a virtual line, in the good direction, then the algorithm increments the correspondent counter (if the current blob is a merging blob, the algorithm looks how many blobs composed it and increments with the same number). Because the algorithm memorizes the features of each blobs it can easily deduce the direction of the motion (for example, by looking the position of the centroid).

The second way is to use just one virtual line which demarcate clearly the IN area and the OUT area. This method was preferred for the final prototype because of its simplicity. To count people, the algorithm just look the position of the bottom segment of each bounding box in two consecutive frames. If, in frame T, the segment is under the virtual line and, in the next frame, the same segment is upper then the algorithm increment its IN count value. But it is really important to use also the bottom segment of bounding box for the OUT count (or else, the algorithm can make error of counting like count several times a person IN without count him OUT; for example a person which make small round trip next to the IN area). The other way to count using one virtual line is to look the position of the centroid of each blobs.

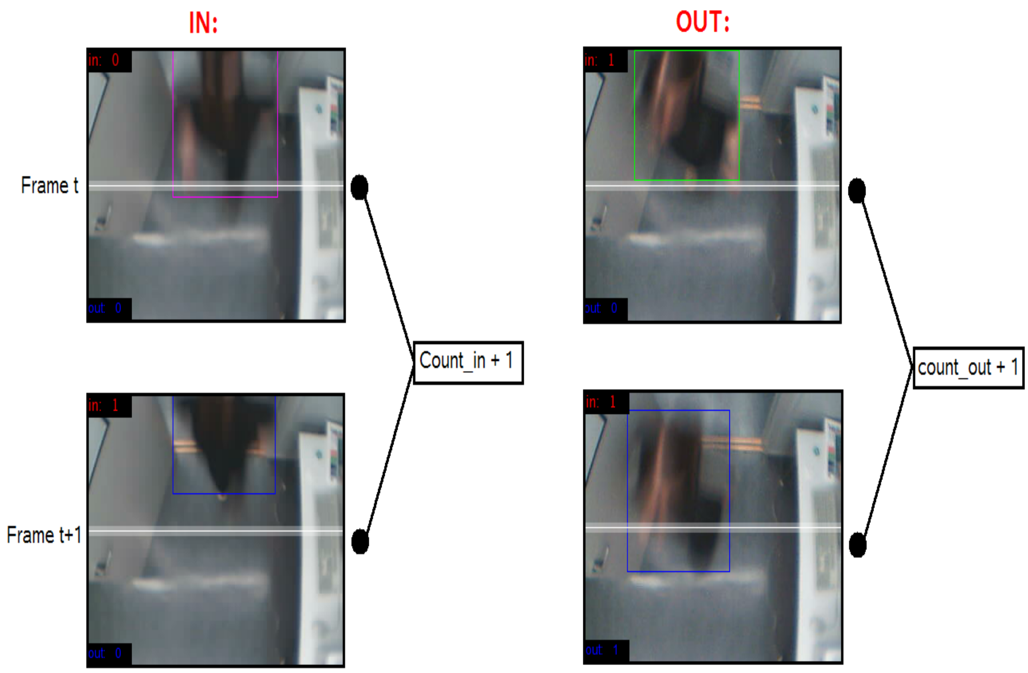


***Illustration 21**: Example of counting with one virtual line*

4 Implementation

This section covers the different choices taken for the people counting system's implementation and the advancement of the prototype.

4.1 Learning of new development tool

My supervisor advise me to use the Matlab-Simulink programming tool because he knew this technology and some basics works have been done concerning the tracking of objects by the Mathworks' team. So before starting the programming, I tried to learn this technology and found very good documentation and tutorials in the official website of Mathworks " http://www.mathworks.com/ ". Learning of Matlab language was not very hard for me because it looks like C language programming and during my 5-year college, I used to program with such language. But I encounter some problems with the Matlab-Simulink technology because it is a totally different approach to make programs (I haven't any knowing about such system). Simulink is a tool for modelling, simulating and analysing multi-domain dynamic system and is widely used in digital signal processing and control theory. It furnished a graphical block environment and a customizable set of block libraries and based on dataflow programming. The advantage of such technology is that it proposed very good tools to see in real-time different behaviours of the simulation when some modifications of situations or parameters occur. The other advantage is Simulink can generate powerful C code.

After this learning process, I study different models of Mathworks more particularly two of them (**viptrackpeople_win.mdl** and **viptraffic_win.mdl**). The first one is able to track multiple moving objects in the field of the camera (detect and memorize moving objects frame to frame) and the second one count the number of cars (in fact moving objects) in highway. I did not encounter problems to understand those two models thanks to a good documentation and a clear code. But I can't use them for my application, I have just taken some ideas. For example, in the tracking people model, there is not background estimation, the first frame of the video is taken as a background and all the segmentation process is done with auto-threshold . And in the cars counting model, the algorithm just count the number of moving objects which are under a virtual line (in fact, the direction of the cars in one side of the highway is the same so no need to detect the car's crossing). Those two models were tested with the two cameras which I used and the result was not very conclusive. The segmentation process was very bad due to the automatic threshold and the different automatic processes of the cameras, as a consequence the tracking process wasn't good too.

4.2 Video cameras

During my work placement, I used two types of video camera owned by my supervisor. The first one was a Creative Live! Cam Voices webcam able to acquire video at 1280x960 Pixels (1,3 Mpixels) and cost approximately 100$. The main problem with this video camera was all their automatic processes which cannot be turned off such as highlight compensation, background noise removal and environmental side chatter removal. All those automatic processes make the work harder particularly for the segmentation algorithm. The second webcam was a VIMICRO USB PC Camera 301x (infra-red camera so very sensitive to light) able to acquire video at 320x240 pixels and cost approximately 20$. The result with this camera was clearly better because it has less automatic processes than the other one (just highlight compensation process).

Illustration 22*: To the left the Creative webcam and to the right the infra-red webcam*

But to avoid the effect of the automatic processes and because of novice skill in Simulink, I choose to mount the camera vertically like Rossi and Bozzoli [ROS94]. It is true that using an overhead camera limits the directions of people entering in the scene (top and bottom side of the image) but remove totally the occlusion problem so make the problem easier to resolve.

4.3 Documentation

This part describes the implementation in Simulink of the different algorithms used to make a people counting system.

4.3.1 People Counting Model

The people Counting Model is composed of four subsystem (Background Process, Segmentation, Tracking and Counting and Display Results). All those subsystems will be describe later. The model uses three global variables which are the Count_In (represents the number of people enter), the Count_Out (represents the number of people exit) and the Connection (represent the socket connection using TCP/IP protocol between the computer which execute the model and the main computer of the building; this connection is done at the launching of the program). The Edit Parameters block allow to fix different parameter such as the position of the virtual line or the minimum and maximum area for each blobs detected. The input of this model is the signal of the video camera, the cycle is repeated for each current frame acquired at 320x240 in RGB24 format. [BG] and [FG] are just flags (like **Label** and **Goto** in other programming language) and represent respectively the existence of the background (boolean) and the presence of moving objects in the foreground (boolean).

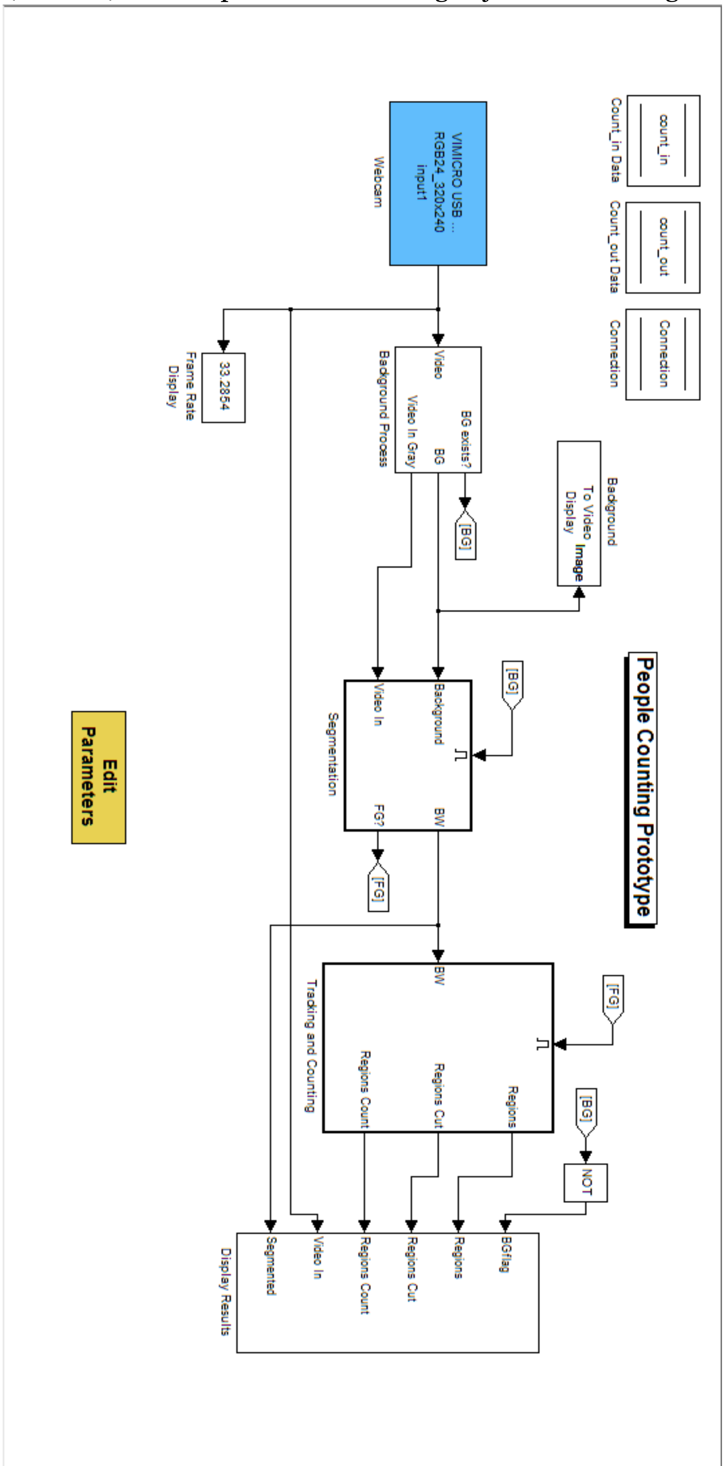


***Illustration 23**: People Counting model*

4.3.2 Background Process Subsystem

The Background Process is composed of two main blocks which are the motion Detector Block and the Background Estimation Block. The input of this subsystem is a colour image (current frame of the video camera). The three outputs are the existence of background, the background image and the current frame transformed in intensity. The Pulse Generator is a block which allowed to create a pulsation (in this case the amplitude of the pulsation is 1 and its length is very small) every each period (the time of this period is customizable).

The Motion Detector block is enabled every each period while the Pulse Generator Block create a pulsation and determine the existence of motion during the current scene represented by its output (boolean). Its input is the current frame of the video transformed in intensity. To detect the motion this block make an absolute difference between two consecutive frames (current frame and previous frame) and find the maximum value of this frame differencing operation. If there is big modification between the two frames (i.e.: the maximum value is greater than a threshold) then there is motion. But in order to have the previous frame, a block Memory is needed. This block allow to memorize the last value of its input (i.e. The last value of the current frame is just its previous frame).

The background Estimation block is enabled when there is no motion on the field of the camera and when the Motion Detector is enabled (i.e.: when the Pulse Generator Block creates a pulsation) and compute the background image. Like this the background will be updated every each period if there is no motion in the field of the camera during this moment. Its input is the current frame of the video transformed in intensity and its two outputs are respectively a boolean which represent the existence of the background and the background image. To compute the background image the algorithm merge three consecutive frames thanks to the Buffer Block and take the median filter. And to know if the background exists (the background is not a black image), the algorithm just look the maximum value of this image and compare to the zero value (if the maximum value is zero then the background does not exist).

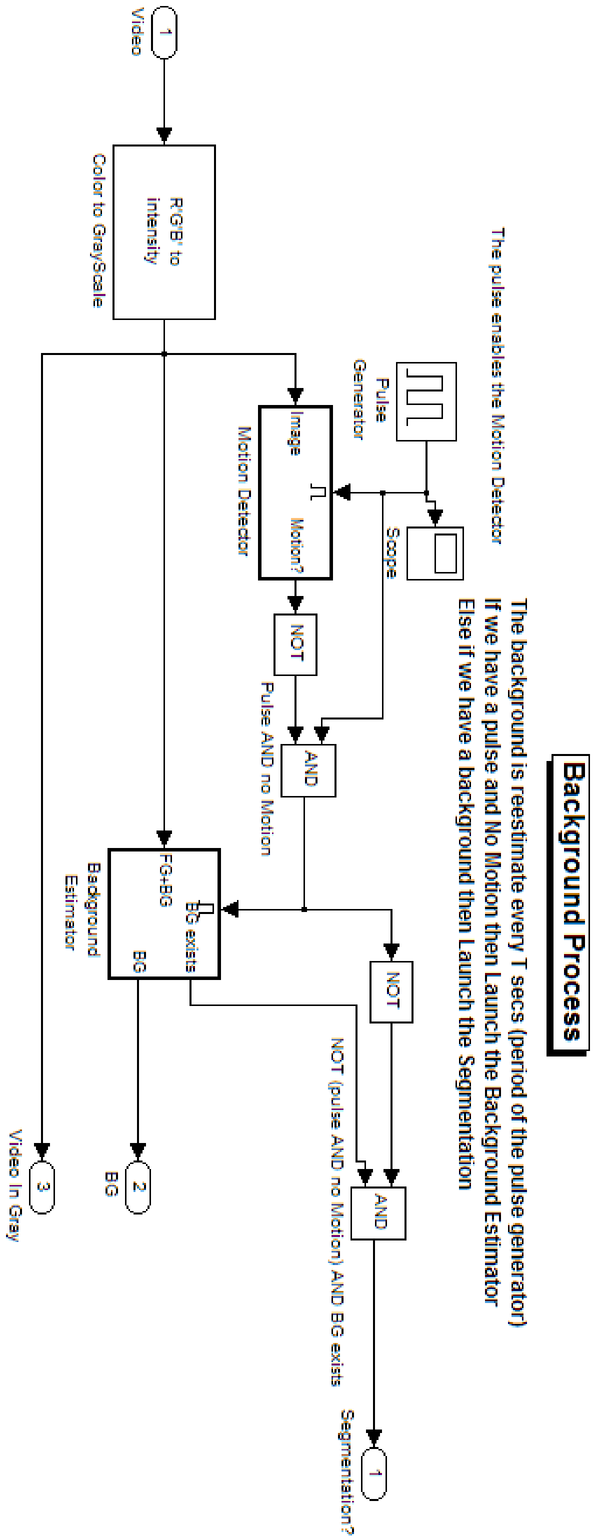


***Illustration 24**: Background Process Subsystem*

4.3.3 Segmentation Subsystem

The Segmentation subsystem is enabled when the background image exists and make a binary image which represent the foreground and the background. Two inputs are needed to make the segmentation, one is the current frame transformed in intensity and the other is the last updated background image. Its outputs are respectively the segmented image (binary image) and a boolean which represent the existence of moving objects (presence of foreground in this binary image). To create the binary image, the algorithm make an absolute difference between the two inputs (background image and current frame). Then two cases are possible, there is a lot of non-zero pixels but closer of zero means that there is no moving objects (this small variations between the current frame and the background is due to the quality of the video camera). And the second case is the presence of big variations means that there is moving objects in the scene. This explains the use of the Switch Block which almost equivalent to a IF THEN ELSE in programming. The Switch block needs three inputs, the input of the middle is the value compare to a threshold (here the sum of each value pixels of the frame differencing image). If this test is positive then the algorithm will take the first input way else it will take the third input way. More clearly, if the sum of all value pixels is smaller than a threshold (no moving objects in the scene) then the frame differencing image will be transformed in a empty image (black image) else the algorithm apply a opening operation to eliminate little blobs due to noise. Finally to make a binary image of this frame differencing image, the Data Type Conversion Block is used.

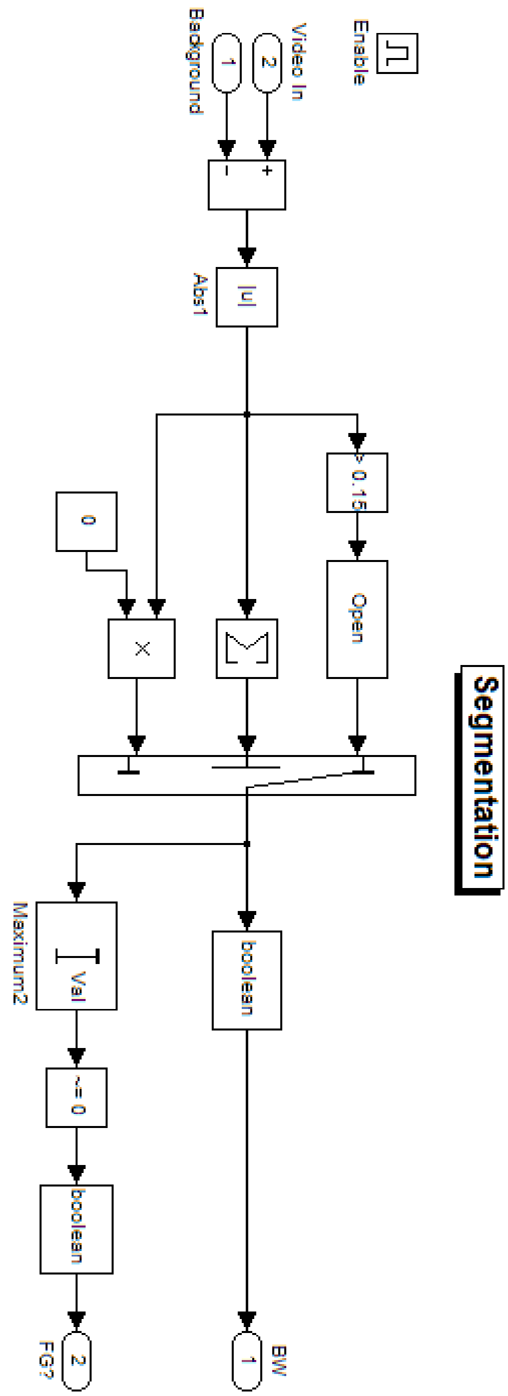


Illustration 25: *Segmentation Subsystem*

4.3.4 Tracking and Counting Subsystem

The Tracking and Counting Subsystem is enabled when there is moving objects in the current frame (i.e.: the segmented image is not an empty image). Its input is the binary image (segmented image) and its outputs are just the different positions of bounding box for the display. Three cases are distinguish, the bounding box will be displayed in Blue the moment when it go through the virtual line and count as a IN or OUT, displayed in Magenta if the bounding box cuts the virtual line and else displayed in Green. The algorithm launch a Blob analysis to detect all the blobs checking the constraints of minimum and maximum areas. To track the different blobs the algorithm need to memorize the position of each bounding box and centroid. In order to know the direction of the motion of each blob, the algorithm compare the position of the bounding box in the current frame with the position of the bounding box in the previous frame. A IN is detected if and only if, the position of the bottom segment of the bounding box in the previous frame is under the virtual line and is upper the virtual line in the current frame. The same “mirror” reasoning is done for the OUT detection. When a IN or OUT is detected then the algorithm increment the good counter and send data to the main computer of the building thanks to the Connection global variable (this process is done in the Add and Send Count Block).

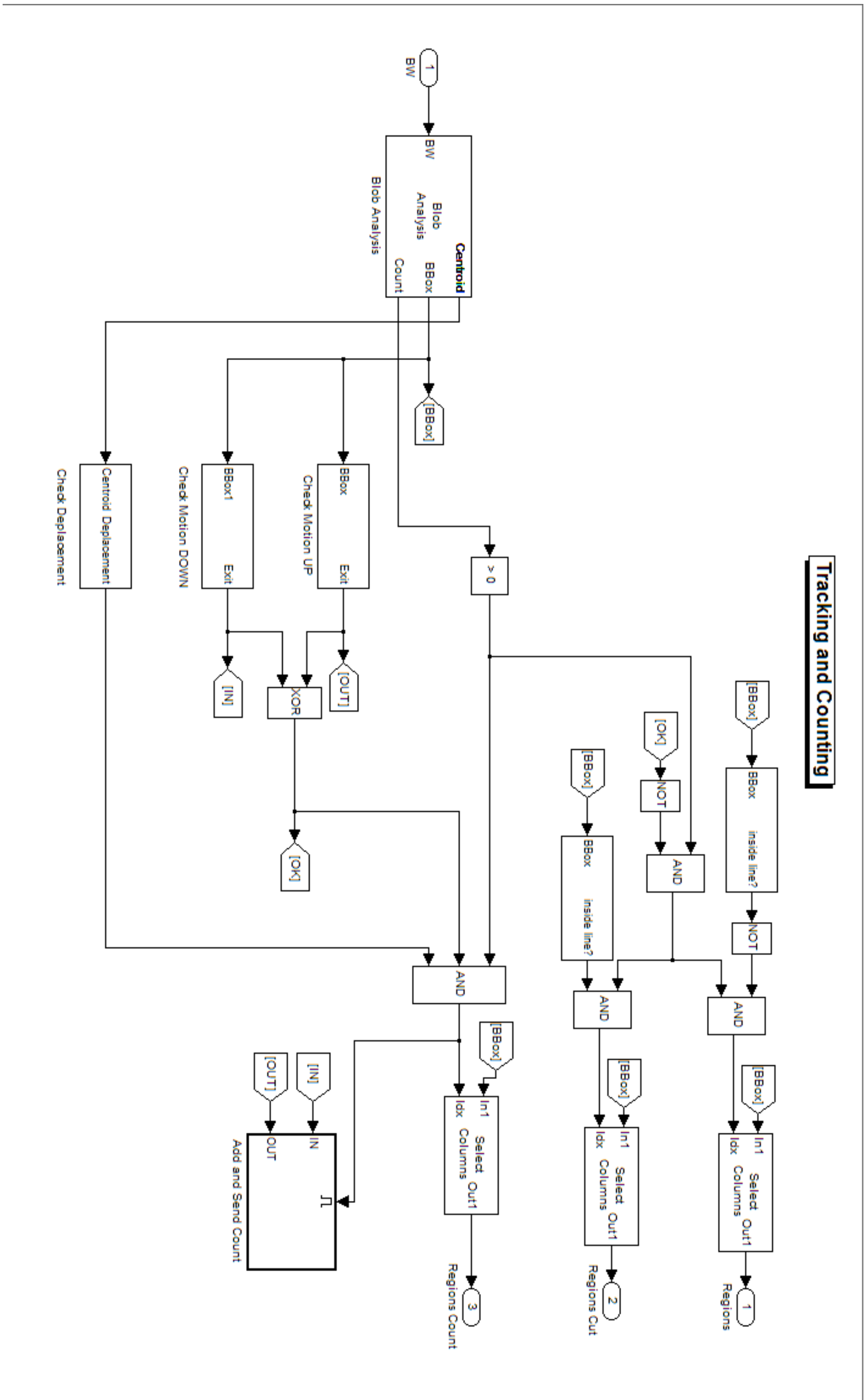


***Illustration 26**: Tracking and Counting Subsystem*

4.3.5 Network Communication with the software

Client

To make TCP/IP connection with my Simulink model, I use a totally free toolbox even for commercial use named TCP/UDP/IP Toolbox 2.0.5 and coded in C by Peter Rydesäter. This toolbox can be found in the Matlab Central File Exchange website http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=345&objectType=File. This toolbox is very easy to use and allow a lot of functionalities necessary for the application. Here are some example of code used in my model:

- As seen before, the Connection global variable is initialized at the launching of the program. Its initial value is “pnet('tcpconnect','192.168.0.1',18000);”. This line of code opens a socket with a TCP/IP connection in the port 18000 and to the computer which has the local IP address '192.168.0.1'.
- In order to send data when a crossing is detected, an embedded Matlab function Block is used. This block allow to add Matlab code in Simulink model. The code for the IN count is “pnet(u,'printf','VideoCam1:1');”. 'u' is the name of the socket and 'VideoCam1:1' is the sent message. VideoCam1 represents the identifying of the sensor (this is useful for the update of the centralised database) and the '1' after the separator ':' represent an IN count ('-1' is the OUT count).

Server

The server program was coded in JAVA using the Eclipse environment (freeware) because it used some other program coded by Yorrick also in JAVA. The server program has to be launched in the main computer of each building, all the sensors (or data logger depending of the type of the sensors) are connected by TCP/IP protocol to this computer. In fact, the electronic students develop a program which can deal with several sensors (just one connection) whereas my model deal with only one camera (it is not possible to launch two people counting models in one computer due to performance). So, because the program must deal with a lot of sensors, its architecture looks like a Server Chat (multi-thread client connections). For each sensor (computer), the program create one **ThreadClient** in order to communicate because different computer can send data to the server in the same time. The algorithm has an another thread called **ThreadQueue**. This thread is launched all the seconds and is used for updating the centralised database with all the informations sent by the sensors during this second.

In order to work correctly, this program uses a configuration file **spring-beans.xml** where several informations are stored like the IP address of the centralised database (in order to communicate with it) and the identification of the building where the program is launched (each building has a unique identification in the centralize database; this part is more detailed in the Yorrick report). At the beginning, the program reads the configuration file and creates an object **Queue** with the current *BuildingID*. This object is connected with the centralised database and gets back all the informations of the current building (Doors and Sensors) thanks to the function of the **Remote Service** Object named **getAllDoorsByBuilding(** *BuildingID* **)**. When a client (a sensor) send a message to the Server, The corresponding **ThreadClient** will analyse the message and update the different counts (IN and OUT for each sensor) which are stored in the object Queue (update with the corresponding *SensorID*). The **ThreadQueue** (which is launched all the seconds) read those different counts and make a connection with centralised database in order to update the corresponding building. If the updating operation is going well, the **ThreadQueue** resets all the count of the object **Queue**.

If you want to understand more, I advise you to read the report of Yorrick which detailed the architecture of the centralised database.

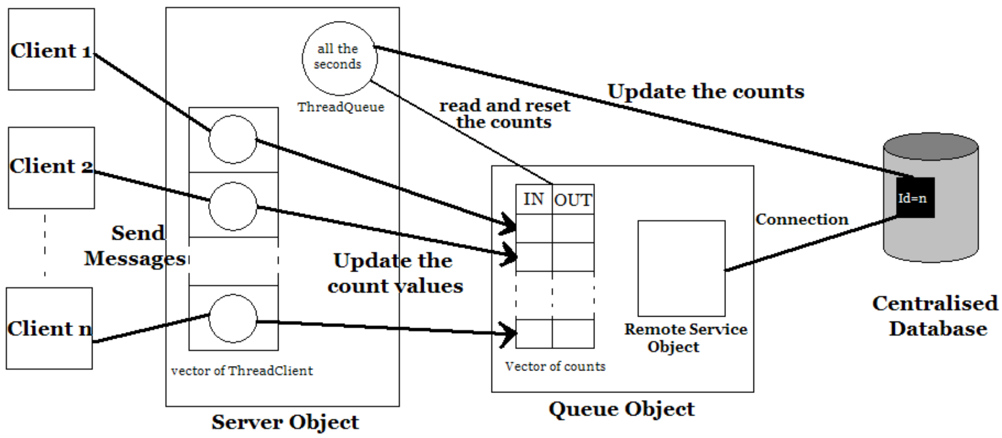


***Illustration 27**: Scheme of the Server program*

5 Results

It was impossible for me to test my model with a truly scientific approach because of the short time and lack of materials. To measure the accuracy of my model, I will have to place the video camera in the entrance of the university during one full day linked to a computer which launch the people counting algorithm. Furthermore, the video camera is a USB webcam so need special USB cable due to the limit of 7 meters of USB protocol. Then, to compare the result, someone will have to count people manually during one full day. But one demonstration of our entire system was down in the end of our work placement to show the advancement to the client. It was at this moment when i could make a real test of my program.

5.1 Set-up and hardware used

During this demonstration, I use the second camera (VIMICRO USB camera) linked to my laptop which launch the Simulink model. Like seen in the *implementation section*, the video camera was placed in overhead to limit occlusions and too big variation of light. The room was spacious and in normal condition of light. My program operated in 15 frames per second with my laptop (Intel Centrino T2300 Dual core at 1,66 GHz and 2 GBytes of RAM). In order to make a demonstration of our entire system as real as possible, we choose to show the system for one building. First, we make a local network between the server (main computer of the building), my laptop (for the video camera) and the laptop of the electronic students (for the laser beam). And for practical reasons, the centralised database was placed in the same room but not integrated in the local network. In order to access to this database, the server communicate thanks to an internet connection (show the fact that the location of the centralised database can be everywhere in the world, not necessarily closer of the main computers). A scheme of our system in the room is present on the next page.

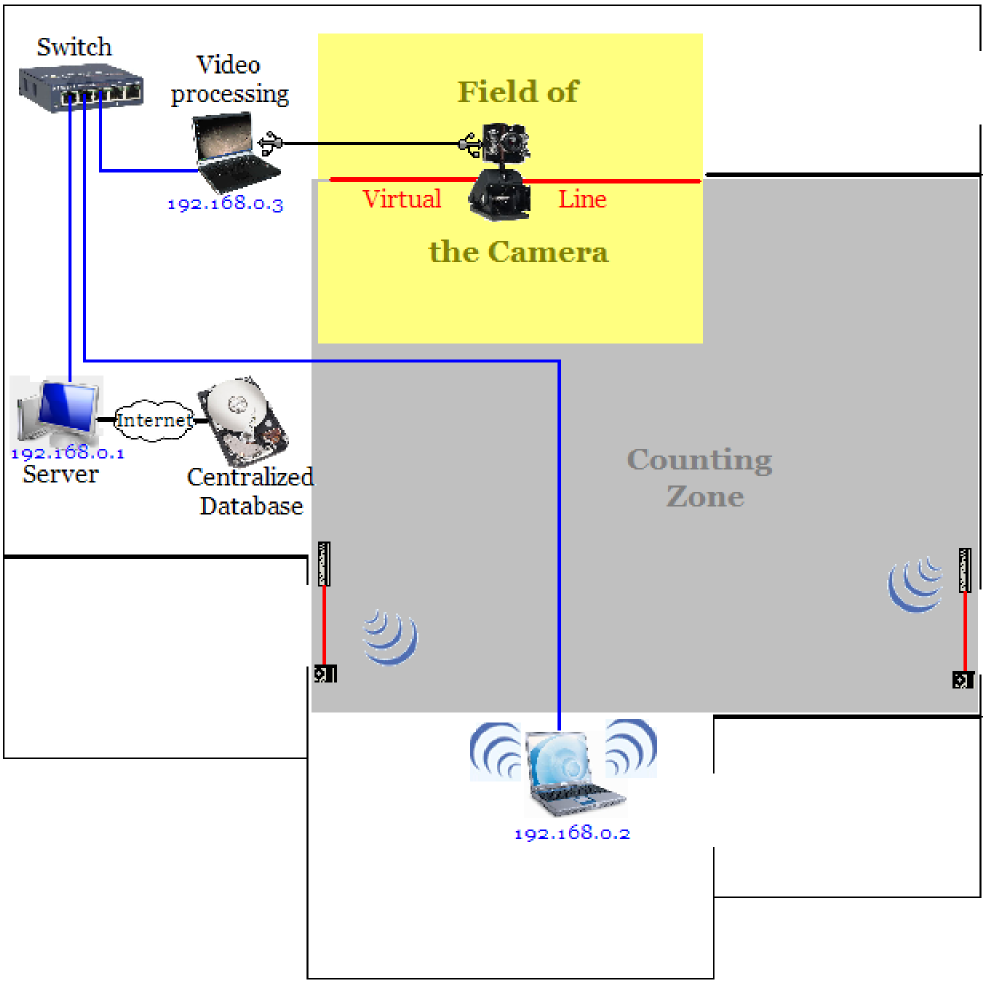


***Illustration 28**: Demonstration Blueprint*

5.2 Tests and evaluations

During this demonstration, we test some situations which could cause problems. The first one was to test my multi-threading count-insertor program. If several sensors detect several persons in the same time and after send the data count to the server (main computer), maybe there will be some loss data. This test was positive, thanks to the TCP/IP protocol and JAVA language which can easily and surely deal with the problem of simultaneous multi-threading. The others tests concern my Simulink code. We knew that laser-beam sensor has problem with multi crossing in the same time. In fact, for example, if two persons cross the horizontal laser beam in the same time (same or opposite direction) then one person will hide the other and the sensor will just count one. Those two different situations were tested and worked in some cases. In fact, the algorithm can deal with multiple crossings if and only if the persons are not too closer (no touch between people) because it is not enough evolute concerning the segmentation and the tracking. All those improvements will be talk later in the *improvement section*. Other tests have been done and was positive too for example very fast and slow crossings. In some case, fast people crossing can cause problem if the number of frames per second of the video acquisition and the speed of the algorithm are not enough high.

So we can conclude that the video camera can be a very good alternative to deal with big entrance (several persons can enter or exit the door in the same time) thus more accurate than other sensor. But the fact is using video camera is more expensive than using other sensors so to reduce the cost of the system, it is not necessary to use video camera for small entrance (normal door); for example the horizontal laser beam is enough.

6 Improvements

Because of the short time limit to develop the people counting system, the prototype is not very advanced and can easily be improved. This section covers all the features which can be added in order to improve the entire people counting algorithm. This section is divided in two parts, the first one describes all the different improvement for a people counting application and the second one gives some ideas to extend the prototype to other applications.

6.1 People counting system

As said in the *Result section* some situations are resolved like the count of several persons which cross a virtual line in the same or opposite direction. But if the persons are too closer (contact), the algorithm is not able to separate correctly those blobs. The first important improvement is about the segmentation. To make the segmentation better, the prototype has to use a better video camera (which haven't automatic process like light compensation), use the information of the motion detector in order to detect occlusions and make a better silhouette of the objects and maybe use colour analysis to classify each pixel in order to remove shadows. Then the tracking module cans be improved by memorizing more informations of each blobs frame to frame like the bounding box, position of the centroid, number of pixel in the area, mean of the pixel gray values. Like this the tracking module will deal, in a better way, the situations of merging and splitting blobs. Finally, the other improvement concerned the counting module. If the program detects a moving objects then launch a human recognition (skin detector, head detector or other modules to detect human).

In the future, this model has to be exported in other language to work with a DSP microprocessor build-in the video camera. Like this, it will reduce the cables and space taken by the current system and make it more commercialized.

6.2 Other applications

A people tracking system cans be used for others applications not necessarily just for counting people. For example, this systems can be extended for security application. In fact, a real-time people tracking system provides enough information in order to make a good video surveillance. Detect strange behaviours of people (like violent gesture, fight or running people) and store those informations on a database. This type of system cans be very interesting for storekeeper or supermarket. An other application is for marketing. It can analyse the behaviours of clients and make conclusion. For example, measure the impact of an advertisement or modification in the arrangement. Determine the period and place of good and bad influence...

7 Conclusion

This master thesis presents an approach to count people passing through a virtual gate using a fixed cheap video camera mounting vertically and Matlab-Simulink programming tool linked to a web application. In *section 5*, the results show that using a camera to count people is good alternative to other sensors for big entrance because more accurate. But it shows also that the system needs a lot of improvements to be really reliable.

This topic was very interesting for me, because it merges a lot of applications such as image and video processing, web application and databases and micro-controller programming. It was good to work in a such various project and discuss about problems with all the person who participated to this project. It permits also to improve my skills in Matlab developing and learning a new development tool Simulink.

But, because of the short time and the time I lost to code with Simulink, I am not very satisfied of my resulting work. I have a lot of ideas of algorithms but I couldn't implement them in Simulink due to my level. I don't regret using Matlab-Simulink tool, but if, one day, I had to continue this project I will restart it from the beginning and use a language with which I am more comfortable such as Matlab for example. I was happy to do my work placement in this great and beautiful country which is Norway, and if I have the opportunity to continue this project then I will return there without any hesitations ^^.

Bibliography

[TER99] : K. Terada, D. Yoshida, S. Oe, and J. Yamaguchi, A method of counting the passing people by using the stereo images, International conference on image processing, 0-7803-5467-2,1999
[BEY99] : D. Beymer and K. Konolige, Real-time tracking of multiple people using stereo, , ,1999
[HAS97] : Hashimoto, K. Morinaka, K. Yoshiike, N. Kawaguchi, C. Matsueda, S, People count system using multi-sensing application, 1997 International conference on solid state sensors and actuators, 0-7803-3829-4,1997
[TES96] : A. Tesei, A. Teschioni, C.S. Regazzoni, G. Vernazza, \Long-Memory" matching of interacting complex objects from real image sequences, 1996 Conference on Time Varying Image Processing and Moving Objects Recognition, ,1996
[SHI91] : A. Shio and J. Sklansky, Segmentation of people in motion, 1991 Proceedings of the IEEE workshop on Visual Motion, 0-8186-2153-2,1991
[SCH95] : A.J. Schofield, T.J. Stonham, and P.A, A RAM based neural network approach to people counting, Fifth International Conference on Image Processing and its Applications , 0-85296-642-3,1995
[SEX95] : G. Sexton, X. Zhang, D. Redpath, and D. Greaves, Advances in automated pedestrian counting, European Convention on Security and Detection , 0-85296-640-7,1995
[SEG96] : J. Segen and S.G. Pingali, A camera-based system for tracking people in real time, Proceedings of the 13th International Conference on Pattern Recognition, 0-8186-7282-X,1996
[HAR01] : I. Haritaoglu and M. Flickner, Detection and tracking of shopping groups in stores, Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition , 0-7695-1272-0,2001
[CON94] : Gary Conrad and Richard Johnsonbaugh, A real-time people counter, Proceedings of the 1994 ACM symposium on Applied computing, 0-89791-647-6 ,1994
[ROS94] : M. Rossi and A. Bozzoli, Tracking and Counting Moving People, IEEE Proc. of Int. Conf. Image Processing, ,1994