Project Spotlight : Report

Jin-Dong Dong, Shih-Yang Su
Department of Computer Science, National Tsing Hua University, Taiwan

1 Abstract

In our project, we propose a system that can perform semantic segmentation to capture the contour of human beings and estimate their coordinates and distances. These informations can help us set off the alarm when some suspicious guy get too closed to a restricted area and so on.

2 Introduction

2.1 Problem Description

Recently, school safety and security has become a great issue, many incidents have taken place in or near school or campus. By improving the current surveillance camera system, we can prevent and eliminate these unpleasant incidents.

So we decide to implement a system that can help us detect human from surveillance camera, and make sure these possible suspects is not too close to restricted area. Once they invade the restricted area, our system can quickly notice their trespassing, and notify security to take action.

2.2 Goals

Our system should achieve the following criteria:

1. Accurately capture human contour
2. Able to approximate the distance/coordinates of human object

(a) The suspicious may intrude your home, campus, etc

(b) A trespass in progress

Figure 1: Figure for problem description and introduction. (a) is taken by *Jake Essman/ The Arbiter*, (b) is from sialicencehub.co.uk

3. Can work in real-time

Firstly, since we want to catch most of the suspects, it is important to capture human accurately from camera frame.

Secondly, since we want to keep those stranger away from protected areas, being able to estimate their distances from those places will be helpful. With this info, we can drive them out even before they enter those areas.

Thirdly, real-time is undoubtedly an essential criteria for this system. If we cannot catch all the suspicious in real-time, and then this system will be totally useless, since the suspicious may get away before we even notice them.

In conclusion, our system should be aimed for meeting the criteria mentioned above.

3 Implementation

As Figure 2(a) shown, we divide our system into 2 parts:

1. Perform semantic segmentation from images input
2. Retrieve coordinates/distance information

By performing semantic segmentation, we can capture human contour, and thus know their exact position in a given image frame. The contour information may be further processed to get even more detailed data, such as pose estimation, or other kinds of feature, these features will be our future works.

On the other hand, retrieving coordinates can help us determine the position and distance of the suspicious.

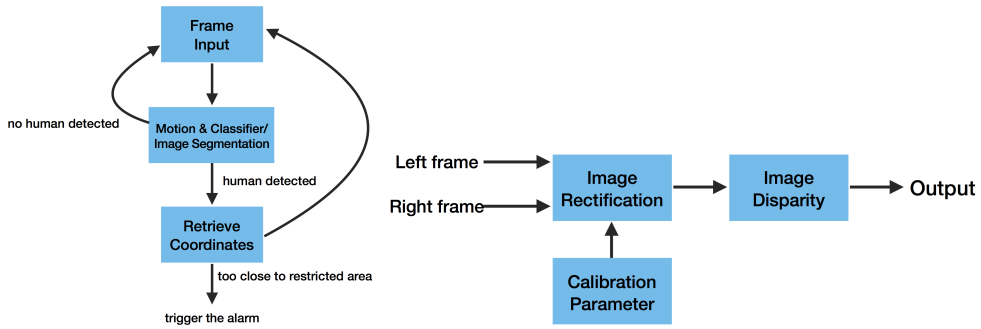
We will discuss about the details of implementation in the following subsections.

3.1 Semantic Segmentation

For semantic segmentation, we will try two methods :

1. ENet : a neural network architecture for real-time semantic segmentation
2. Motion tracking combines with image classifier.

In the following subsection, we will discuss about our choices in details.



(a) System flow (b) Flow of retrieving coordinates

Figure 2: Flow of system(a) and retrieving coordinates(b)

(a) Before segmentation (b) After segementation

Figure 3: Example of segmentation. Photo created by Tao Zhao and Ram Nevatia

3.1.1 ENet : Semantic Segmentation

ENet is a newly announced neural network architecture which aimes for performing real-time semantic segmentation, and it has been proven to be capable of running in real time and work sufficiently on embedded system such as Nvidia TX1.

However, ENet is not famous for its accuracy. Although its encode-decode structure can be very robust in terms of computing speed, it may not generate the accuracy we want to pursuit. But we still consider it worth a shot to try this architecture in our system.

3.1.2 Motion Tracking & Image Classification

The other method is to combine motion tracking and image classification. The workflow is show in Figure 2. In this method, we only interest in identifying a currently moving object, the assumption is that if the suspicious want to trepass a certain area, they must be moving. So we only crop the area that has moving object, and identify if that area has a human object. Once we confirm that the moving object is a human, we can calculate their positions.

There are two advantages of using this method:

- Less computation time(not sure?)
- Can be more accurate

Since we only need to perform detection task on a small region, the overall computation

(a) Apply motion tracking (b) Classify object

Figure 4: Example of motion tracking & image classification

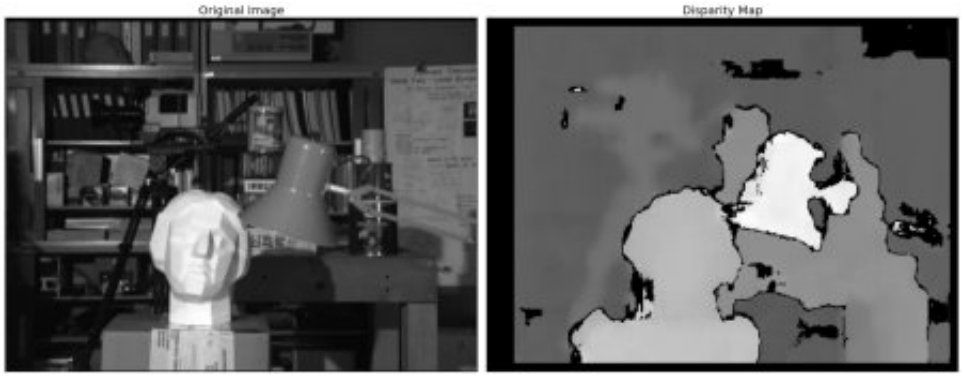


Figure 5: Example of disparity map. Picture from *OpenCV official document*

should be less then the one in Section 3.1.1. Secondly, since we already know the position of the object, we can directly use a well-trained robust image classifier to identify the object.

However, the drawback of this method is that if the object is moving too fast or too slow, this method may fail to notice the change of image frame, and thus fail to detect human object.

3.2 Retrieve Coordinates

Figure 2(b) show the flow diagram of retrieving coordinates. To accquire the coordinate of human object we obtained from Section 3.1, we first need to get a stereo image. A general approach to get a stereo image is to use stereo camera. In our system, we combine two cameras, and impose some post-processing technique to reproduce the effect of stereo camera. After the stereo image be correctly generated, we can calculate the disparity, and retrieve the depth and coordinate information afterward.

3.2.1 Camera Calibration

To create the effect of stereo camera, the first step is to compute the intrinsics, extrinsics and distortion coefficientes of both cameras. We use OpenCV library to help us compute these calibration parameters.

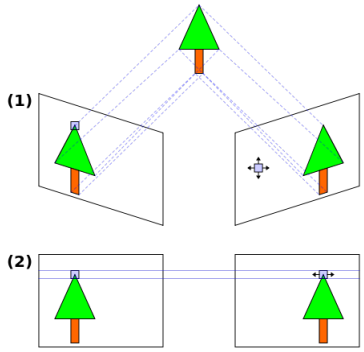


Figure 6: Example of image rectification. (1) before rectification (2) after rectification. Picture from *Wikipedia*

3.2.2 Image Rectification & Disparity

With calibration parameters obtained from Section 3.2.1, we can perform image rectification. The purpose of image rectification is to compute a matrix of coefficients, which can be used to transform multiple images onto a common image plane(see Figure 6). With these coefficients, we can successfully generate a stereo image with depth information, that is, a disparity map(see Figure 5) from our hand-made cameras. While there are some algorithm designed for computing disparity map, we choose *Semi-Global Block-Matching Algorithm*(SGBM) to compute it, since SGBM is more efficient then brute-force algorithms, such as Block-Matching.

Same as Section 3.2.1, we also use OpenCV here to complete the task.