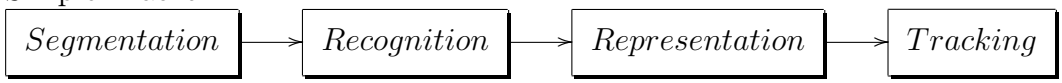
Face recognition and tracking

Lluís-Pere de las Heras Caballero Ahmed Mounir Gad
Mónica Piñol Naranjo

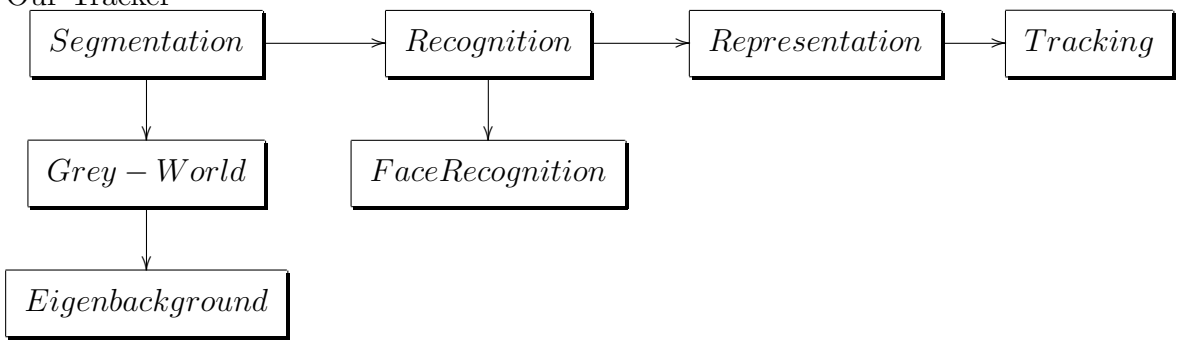
March 16, 2010

1 Introduction

- Simple Tracker



- Our Tracker



In this project our main focus is to build a single and multiple target tracker. We extend this approach to build a useful real-life application to that tracker which is a face recognizer and tracker.

Our project first starts the normal tracking process which is segmentation and only determining the useful parts of the scene. We use these useful parts of the scene to detect the faces and correspond these faces to the interesting faces in the scene. We afterwards calculate the velocity of the person having this face and we keep tracking until the person leaves the scene or our tracker fails.

2 Segmentation

The process of segmentation is to separate the background to foreground.

- The Background subtraction use three parameters: **gamma**,**tau** and **radius**.

$$B_{i+1} = \gamma * F_i + (1 - \gamma) * B_i \tag{1}$$

  - Gamma is the learning rate, usually is 0,05.
  - Tau is the different between frame to background.
  - Radius is the parameter to delete the little blobs.

- The *background subtraction* is not robustness to change illumination, for this we implement *Grey-World* because is invariant to illuminant changes. The Grey-World method try to eliminate the grey in the image, for this reason move the minimum and the maximum values.

$$(R'G'B') = \begin{pmatrix} \frac{R_{grey}}{R_m(I)} & 0 & 0 \\ 0 & \frac{G_{grey}}{G_m(I)} & 0 \\ 0 & 0 & \frac{B_{grey}}{B_m(I)} \end{pmatrix} * \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{2}$$

- The *selectivity* is the other method that use the simple different between foreground with background but the selectivity a prior knows that pixel is or isn't foreground.

$$\begin{cases} B_{i+1}(x,y) = \alpha * F_t(x,y) + (1-\alpha) * B_t(x,y) & if F_t(x,y) is Background \\ B_{i+1}(x,y) = B_t(x,y) & if F_t(x,y) is Foreground \end{cases} \tag{3}$$

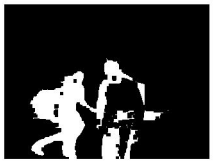
- The *eigenbackground* method use the Principal Component Analisys (PCA) to reduce the dimensionality. Also, the first M eigenvectors are the eigenbackground.

  eigenbackground = eigenvectors(:,1:M)
  Foreground = I - I'
  I' = Image Projection and Reconstruction (Background)

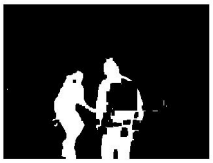
3 Detection

Now the segmenter has finished its work and we have a number of blobs that correspond to the changes in the background. Some of these changes are definitely the objects we are interested in. This step in the tracker mainly focuses on detecting the important parts of these background changes.

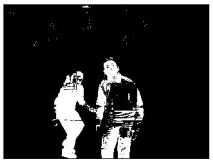
The basic detector provided in our framework is based on simple blob detection. It just returns each set of connected blobs as an object. We basically have two questions to investigate about. The first is how the basic tracker provided behave in the presence of noisy detection. The second is



(a) Background Subtraction



(b) Selectivity



(c) Eigenbackground

(d) Frame 300

Figure 1: Different segmentation with the same frame

how the tracker behaves in the case of no detection. The answers to these two questions are illustrated in the tracking section.

We did several optimizations to the simple detector provided. The first was to detect faces and keep tracking those detected faces. We thought of an interesting application that may use this technique. We added a face recognizer that only detects faces of people of interest to the tracker. We provide a trained Support Vector Machine(SVM) classifier to our tracker. We train it using labeled faces detected from a video for 5 persons. The detector then will only detect the people of interest in any video and the tracker will only track those people.

In case a match was found, the detector will return the whole blob that made that match. Optimally, this blob should be the whole body of the person. Based on the information from the segmenter, the moving object having the detected face should be the whole person. We then start tracking this person even when the detector fails. A part of the method for doing this is discussed in the representation and the rest is in the tracking section.

4 Representation

5 Tracking

In this section we will discuss the main module of the project which is the tracker. A simple Kalman filter tracker has been provided in the tracking framework. It uses results of the representer to track the position and the extent of the object being tracked. The Kalman filter works by estimating an unobservable state which is updated in time with a linear state update and additive Gaussian noise. It uses the technique described in [1].

6 Conclusion

References

[1] Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2001.