Extract feature points from faces to track eye's movement

August 13, 2009

Peter Aldrian, Uwe Meier, Andre Pura

Students at the University of Leoben

Abstract

This paper handles with the question how to extract fixed feature points from a given face in a real time environment. It is based on the idea, that a face is given by Viola Jones Algorithm for face detection and processed to track pupil movement in relation to the face without using infrared light.

Contents

1 Introduction and Idea

There are a lot of projects running all over the world trying to track faces and points in it. Our target is to track eye movement in correlation to other fixed points in the face without using infrared light. Infrared tracking of pupils is the current state of the art, but it has some disadvantages as it can cause damage on eyes.[1] Further it is necessary to have infrared light, which can be reflected in the eyes.

In contrast to infrared light tracking, it is tried to follow eyes movement by extracting different features from a standard webcam picture and process the given data in a real time environment.

2 Environment

It is necessary to define two different environments. First of all the real world, secondly the programming environment. The first one defines where the webcam actually is, the second one, where the system is running on.

2.1 Real world environment

There are a lot of influences, which can manipulate the result of tests in a significant way. All tests are made under "normal" indoor light conditions, which mean that there is no extreme light coming from sides. Of course, there must also be enough light in the room, otherwise a visual tracking can not work. Best results are made with light from above.

2.2 Programming environment (Soft- & Hardware)

For tests and first implementations Matlab R2009a[2] was used. It is not as fast as C code id, but not fast enough for real-time experiences. Another advantage of Matlab on Windows machines is the possibility to simple access webcams with the Image Acquisition Toolbox[3].[1] On the hardware side a Logitech Quickcam Pro 9000[4] is used for image acquisition and a Lenovo ThinkCentre 639471G[5] (Intel® Core 2 Duo E7200 Processor, 2 GB RAM, Windows Vista) for testing.

3 Feature extraction

First of all the face is detected by a modified Open CV's Viola Jones implementation.[2][6] As figure 1 shows, it is possible to extract areas from Face's

[1] The Image Acquisition Toolbox is not available for Unix based OS yet.
[2] Memory leaks were fixed.

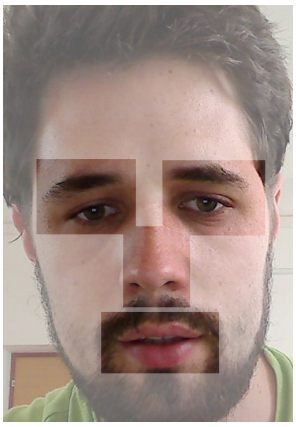

Figure 1: Face Geometry, highlighted areas are the search area for different features, e.g. eyes, nose, nose holes, mouth.

geometry, where the points should be. These areas are the inputs for the exact search functions described below in 3.1, 3.2 and 3.3.

3.1 Nose holes

Finding nose holes in an area given from face's geometry depends on the angle between camera and face. If there isn't a direct line of sight between nose holes and camera, it is obviously not possible to track them.

Nose holes' color have a significant saturation, depending on its color black. The threshold must be defined and over geometry or clustering two centers of saturation can be found.

3.2 Mouth

Detecting the middle of the mouth isn't as simple as it is thought. There are a lot of possibilities, going over gradient horizontal and/or vertical decent, hue or saturation. At the moment it is implemented utilizing the distinct hue of lips. Light reflects on lips and this point is fetched by a defined hue value. In contrast to the methods, this method is not light independent, thus intensity and direction of the light can influence results. A better method should be included in the future.

3.3 Eyes and pupils

A lot of ways can be developed to find pupils in the given area surrounding the eyes. It can also be done using hue or saturation, which leads - controlled

Figure 2: Picture of a pupil, which is used for calculation

Figure 3: Getting the Pupils by Anirudh S.K.'s algorithm

conditions given - to good results, but it highly depends on the current light situation.

Thus, another way is used to find the pupils. A picture (See figure 2) of the pupil runs over the current picture area and tries to find the place with the highest accordance. Different pupils where used for testing and the best results were gained by pupils directly from the tester, which was not really surprising. Obtaining them is not that simple though. An algorithm from Anirudh S.K. called "Integrodifferential operator"[7] was used, which requires too much calculation time to be used in real-time environments, but is fast enough for getting the first pupils (See figure 3) .

4 Results

The results are quiet good, but they are depending a little bit on light conditions and the used camera. With described test system it runs with 2 fps (See table1). The camera is able to give 4 fps in maximum resolution.

Operation	**Time [s]**
Get picture from Camera	0.2
Process image	0.3
Total	**0.5**

Table 1: Elapsed time per cycle

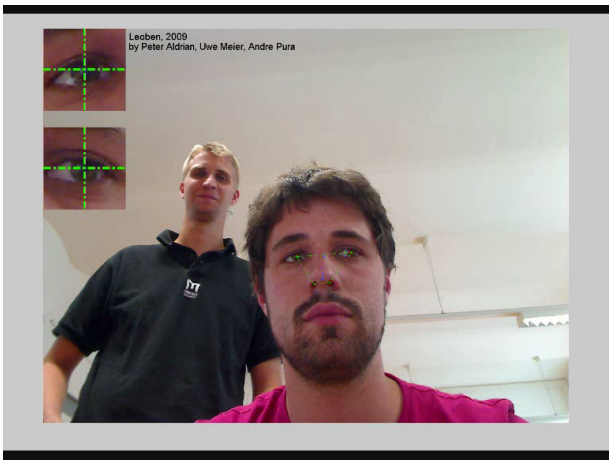

Figure 4: Example 1

Some more image with marked features:

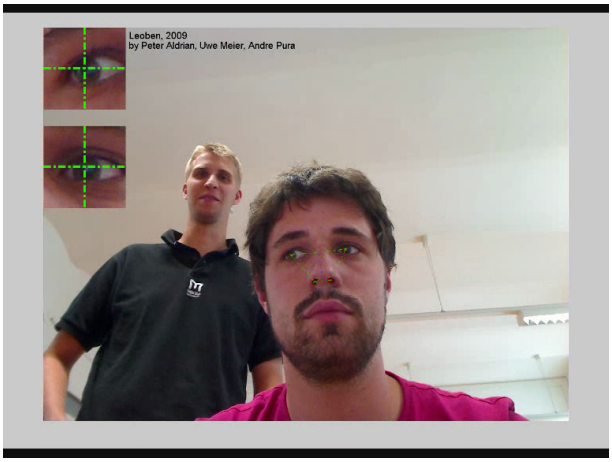

Figure 5: Example 2

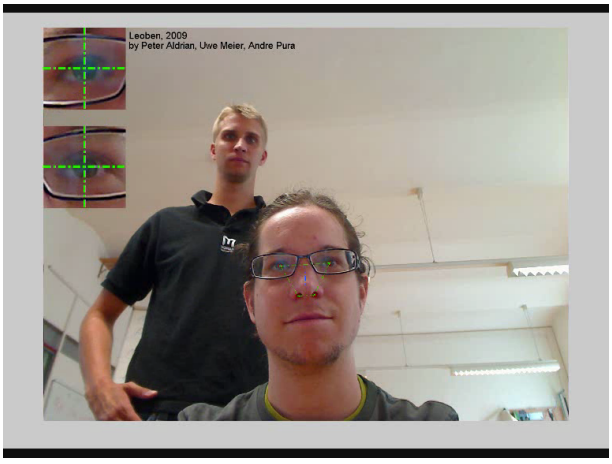

Figure 6: Example 3

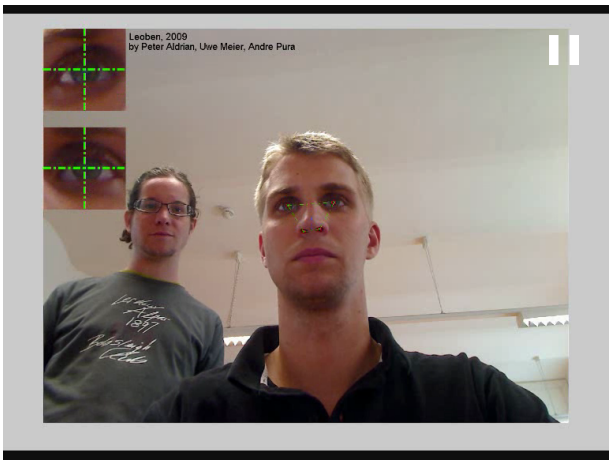

Figure 7: Example 4

References

[1] http://en.wikipedia.org/wiki/Laser_safety , Infrared Light

[2] http://www.mathworks.com/products/new_products/latest_features.html , Matlab R2009a

[3] http://www.mathworks.com/products/imaq/ , Image Acquisition Toolbox

[4] http://www.logitech.com/index.cfm/webcam_communications/webcams/devices/3056&cl=DE,DE , Camera

[5] http://www5.pc.ibm.com/de/products.nsf/$wwwPartNumLookup/_SE271GE?OpenDocument , Test Computer

[6] http://www.mathworks.com/matlabcentral/fileexchange/19912, Open CV Viola-Jones Face Detection in Matlab, by Sreekar Krishna

[7] http://www.mathworks.com/matlabcentral/fileexchange/15652

Peter Aldrian
peter.aldrian@gmail.com

Uwe Meier
uwemei@gmail.com

Andre Pura
andre.pura@gmail.com