Investigate the Challenges of Mobile Eye-Tracking

Alberto Muñoz Ciáurriz

14th - June - 2012

Acknowledgements

It seems like yesterday, but it's almost been ten months. Ten months since I started my final project as an Erasmus student at IT-University at Copenhagen. It has proved to be one of the best decisions I have made so far in my life. I'd probably have to write another report just to express what I've experienced here. I've been joined by many people in these journey, which started many years ago and ends now, and I would like to thank some of them in these lines.

First, I would like to thank my supervisor here at ITU, Dan Witzner. You didn't know me, but that didn't stop you welcoming me, guide me through this project, pushing me further with every new idea you had. You've showed that there's more than meets the eye when dealing with eye tracking devices. Also, thanks to Arantxa Villanueva, my supervisor back at UPNA. You've been the link with UPNA, and the balance between responsibility and enjoyment. Thank you for allowing me to live this experience. Special thanks to Sonia Porta, without all your fighting against all odds (or administration personnel) this wouldn't have been possible at all.

Second, thanks to all the people I've met these months in Copenhagen. Betty, you welcomed me into your "Danish family", one that I like to think I'm part of now. Also, it gave me the chance to meet many individuals that now I call friends (Jonas, Marcin, Kenni, Javier and many more). Thanks too to my flatmates, specially Josh, Zanny and Annelie. These past months have been awesome, but I seriously need a Daim detox straight away. Big thanks to Laura. You arrived at the right time. Your help, our talks, our coffee breaks, have all made this last stage of the project way more bearable.

Big big thanks to all my friends back in Spain, both Telecos and non-Telecos. To the Telecos, thanks for all those uncountable library hours rivalling with uncountable coffee breaks, thanks for the bbq's, for the many dinners and nights out. Without all of you, I wouldn't have finished this. To the non-Telecos, thanks for trying to understand my sometimes erratic behaviour, for all those movie nights, or mac-auto's trips.

Also, I would like to specially thank Sada. You've listened to me, you've helped, always encouraging me not to quit, to keep going. That's what I call a Friend. Thanks also to Ignatius, for teaching me how to compromise, how to unwind when exams were close. Also, a Friend. To Sara, for all those Cantine chats, good advise and nice sightseeing experiences. One more Friend. To Marta, for all your incredible notes, for your carpooling services and for being there when in need, another Friend. To Nune, these last stages would not have been as good without sitting side by side with you at the library. Another Friend! Last, but not least, to Torres. Since day zero you've been there, no strings attached, sharing life experiences and advices among other things. True Friend.

There are more friends I'd like to dedicate some lines, but this would be endless. Special thanks to Libe, Ainara, Txus, Cesar, Eli, Silvia, Min, Zoco, Txiki, Johannes, Belén, Olier, Roncal, Miri, George, Lourdes, Guindano, Natalia, Sagi, Mozo, Troyas, Tximi, Ii, Maider, i-Noa, Aitor, and many more that I'm sure I'm forgetting).

And with all due respects, the biggest thanks go to my family. To my little sister, for all those moments of laughter followed by crying or chasing each other around the house, watching TV-Shows or impersonating other family members. Priceless. And to my parents. For coping with failures, for trying to understand what my degree was about so hard (I think you still don't know for sure, neither do I though). For making the effort of sending me abroad. And "just" for being there. Thank you very much.

Y con todos los respetos, el mayor agradecimiento es para mi familia. A mi hermanica, por todos esos momentos de risas seguidos de llantos o persecuciones por la casa, viendo series o imitando a otros miembros de la familia. No tienen precio. Y a mis padres. Por aguantar todos los fracasos, por intentar entender con todas sus fuerzas de que iba mi carrera (de hecho creo que aun no lo habeis terminado de captar, pero tranquis, que yo tampoco). Por hacer el gran esfuerzo de mandarme al extranjero. Y gracias, "sólo" por estar ahi. Muchas gracias.

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 General Overview 2

2 Theoretical Principles **3**

2.1 Introduction 3

2.2 Homography Principles 4

2.3 Harris Corner Detector 6

2.4 RANSAC 8

2.5 Shape Context Principles 9

3 Experiment **13**

3.1 Design Parameters 13

3.1.1 Matlab 13

3.1.2 Arduino 14

3.1.3 LED-Driver 14

3.1.4 Light Sources 14

3.1.5 Camera 15

3.2 Settings 15

3.2.1 Matlab - Arduino 15

3.2.2 Arduino - LED driver 15

3.2.3 LED driver - LEDs 17
3.2.4 Shapes 17
3.3 Scenarios 20
3.3.1 Points of Analysis 21
3.3.2 Test 1 21
3.3.3 Test 2 22
3.3.4 Test 3 23
3.4 Data Acquisition 25
3.4.1 Homography 26
3.4.2 Shape Context 29
3.5 Analysis 31
3.5.1 Homography 33
3.5.2 Shape Context 39
3.6 Results 50

4 Conclusions **55**

List of Tables

3.1 Position in the scenarios 22
3.2 Distances for array and support arm 22
3.3 Distance from Support Arm to Edge on Test 2 22
3.4 Number of pictures taken for the tests 26
3.5 Values for varying ”w” and ”dmax” parameters 34
3.6 Test 1 - Homography Percentages 34
3.7 Test 2 - Homography Percentages 35
3.8 Test 2 - Mean of percentages for each user 37
3.9 Test 3 - Percentages 38
3.10 Test 3 - Mean of percentages for each shape 38
3.11 Test 1 - Error Percentage 40
3.12 Test 1 - Bending Energy 40
3.13 Test 2 - Medium error (percentage) per user 42
3.14 Test 2 - Medium bending energy per user 43
3.15 Test 2 - Mean error per shape 46
3.16 Test 2 - Mean bending energy per shape 47
3.17 Test 3 - Medium error (percentage) per user 48
3.18 Test 3 - Bending enery (percentage) per user 49
3.19 Test 3 - Mean error per shape 49
3.20 Test 3 - Mean bending energy per shape 50

List of Figures

2.1 Point **x** on plane **Π** is projected to point **x'** on plane **Π'** using homography **H** 5

2.2 Homography - Fundamental Tranformations. Taken from Hartley [6] 5

2.3 Corners on different shapes 6

2.4 Example of two windows used for Harris Corner detection . . 7

2.5 RANSAC applied to a set of points 9

3.1 Schematic for connecting the Arduino board and the MAX7221 16

3.2 Connections between Arduino Uno and MAX7221 16

3.3 Schematic for connecting the LEDs to the MAX7221 17

3.4 Capture of the connections between MAX7221 and the LEDs . 17

3.5 Edge Shape vs. Full Shape 18

3.6 LEDs shapes in the array 19

3.7 Shapes from the "ideal" database created with Matlab 20

3.8 Points to look during the tests 21

3.9 Eyes of the different users 24

3.10 Outdoor shot of the eye 24

3.11 Pictures from the Homography method. 30

3.12 Pictures from the Shape Context method. 32

3.13 Test 1 - Homography Percentages 34

3.14 Test 2 - Homography Percentages (mean value) 35

3.15 Test 2 - Performance of the shapes on different users 37
3.16 Test 2 - Mean percentage per user 38
3.17 Test 2 - Performance of the shapes on different users - Error Percentage (%) 42
3.18 Test 2 - Performance of the shapes on different users - Bending Energy 44
3.19 Test 2 - Mean error (%) per shape 46
3.20 Test 2 - Mean bending energy per shape 47

Chapter 1

Introduction

1.1 Motivation

Eye tracking: the process of measuring the point of gaze or the motion of the eye. Behind this definition lies a technology that has been evolving for more than 20 years. Its applications range from marketing and psychology research to substituting the mouse on a computer. Development of the technology has been accompanied by an increase of people using it. Therefore, at the same time that old difficulties where overcome, new challenges were created.

Eye tracking can be divided into two groups, depending on how it is used: mobile and non-mobile. A non-mobile setup is a rigid one, with the camera usually placed in a fixed position, in a controlled environment and normally requires calibration before being used. On the other hand, mobile setups are more flexible. For instance, the camera doesn't always have to be in a specific location: it can be sitting on top of the computer screen, or under it, on a support arm, or head-mounted. These kind of gaze trackers are becoming more popular, as they permit more flexibility of use.

The consequence of this flexibility is they have to work in environments with changing conditions, so it would be good thing for them not to require calibration every time. This leads to the search of a system, a method that is robust enough to work under varying conditions such as light conditions (number and position of the sources, intensity of the light), indoor or outdoor application, weather conditions (sunny, cloudy). At the same time, that robustness should include the potential users of the system: skin colour, eye colour, morphology of the eye, etc.

The aim of this project is to find a solution robust enough to handle the issues mentioned before. Using methods that are already employed, improving them and changing their parameters; taking methods from other fields an applying them to eye tracking; combining both old and new ways of tracking the eye. It could set the foundations for a new line of investigations and development in the area of mobile eye tracking.

1.2 General Overview

This thesis is structured as it follows:

First, in Chapter 2 the two main methods employed in the project are explained. Also, this chapter explains the sub-methods required for the correct performance of the main ones.

Second, in Chapter 3 explains how the experiment was done: choosing the programs employed, setting everything up, describing the scenarios analysed. Also, how the data was acquired and analysed. In the end, the results are explained.

The conclusions, discussion and future work are all described in Chapter 4.

Chapter 2

Theoretical Principles

2.1 Introduction

In every person's life, the eyes becomes a key organ for perceiving the world. Reading, watching TV, looking to the sides before crossing a road. All of these are essential tasks in which the eyes are crucial. Understanding the visual system, how everything works from the point the eye focuses on something until the brain sends the order to react to what it has seen, has become a main field of research.

Under this premise, different technologies started to appear as a result of these research and investigation; eye tracking and gaze tracking are among them. These technologies allow the monitoring of the point where an individual is looking. Applications range from medical research and human-computer interaction to marketing and advertisement. These applications can be separated into two categories, as Duchowsky [3] classified: diagnostic and interactive.

Diagnostic applications gather data that comes from measuring eye movements when a person is reading a book, watching TV, driving a car or staring out of the window. After gathering it, it's processed; therefore, where the eyes are looking at has no effect. Research, marketing and psychology are among some of the field in which this has use.

Interactive applications process the information coming from the gaze tracker in real time. This way, the gaze tracker can be seen as an input device. An application that is very useful is substituting the mouse of a computer. As a result, a lot of effort is being put into this area, improving the systems already in placed, reducing their costs (as can be read in [8]).

A vital part of both these applications is eye detection and tracking. As mentioned in [5], research in this area focuses on two areas: eye location in the image and gaze estimation. This project focuses on finding more robust methods for eye detection and tracking.

The novelty resides in using patterns of light instead of single points. An array of LEDs will be built and different shapes will be lit. This will create a glint on the eye's surface, which will be different in size from the ones usually used. These new ones will be bigger and will have a distinct shape. In order to make this work, to recognise the shaped-glints, a not-so-common approach is taken: employing object recognition methods for detecting the glints.

These shapes are the ones used for determining where the person's gaze is as a 2D point on a plane (Point of Regard, PoR). The main issue then is to find those glints generated by the shapes and calculate a proper mapping of the work done. The present project focuses on this: finding the shapes accurately so they could be used for mapping.

It was chosen not to do any calibration of either the camera or the hardware. That is why one of the methods employed is Homography, in order to calculate the mapping between the shapes and the screen . The other method this project relies on is Shape Context matching, as it is a method for recognising objects widely used. In the following sections of the chapter, both will be explained.

This Chapter will first explain one of the main methods employed: Homography 2.2. Next, it will explain two methods that were used for preprocessing the data before calculating the homography, Harris Corner Detector 2.3 and RANSAC 2.4.

2.2 Homography Principles

Homography, and more specifically, planar homography, is a projective transformation that maps points form one plane to another plane. This makes it possible for a point **x** on a plane **Π** to find a match point **x'** on a plane **Π′** through a 2D homography (Figure 2.2b). The equation is simple, as can be seen in Equation 2.1. Figure 2.2b illustrates this matter.

$$x' = Hx \tag{2.1}$$

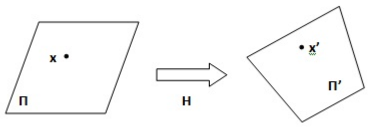


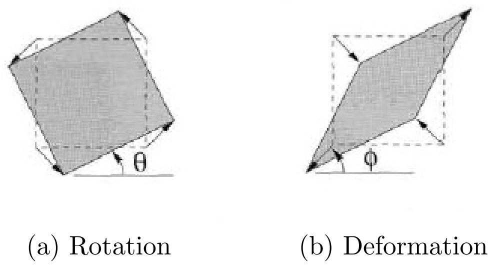
Figure 2.1: Point **x** on plane **Π** is projected to point **x'** on plane **Π'** using homography **H**

Homography **H** is a 3x3 non-singular matrix with 8 degrees of freedom. 4 points or more on one plane and their corresponding 4 or more points on the other plane are needed for relating both planes (and the points) through the homography **H**. If the matrix is examined more closely in equation 2.2.

$$x' = Hx = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{V}^T & v \end{bmatrix} x \tag{2.2}$$

Matrix H (Equation 2.2) can be analysed and it confirms the fact that a projective transformation is a generalization (under certain conditions) of an affine transformation:

- "**A**". Responsible for rotations and non-isotropic scaling. Figure 2.2 illustrates this.



(a) Rotation (b) Deformation

Figure 2.2: Homography - Fundamental Tranformations. Taken from Hartley [6]

- "**t**". Responsible for translations, in X-axis and Y-axis.
- "$\boldsymbol{V^T}$". Responsible for the non-linear effects of the projectivity.

In the following sections, Harris Corner Detector 2.3 and RANSAC 2.4 will be explained. These are two methods that were used for processing the data, prior to calculating the homography.

2.3 Harris Corner Detector

Once the glint/shape is detected, it is essential to extract feature points from it. There are many kinds of spatial feature detection: corner detector, edge detector, radial feature detector, etc. For working with the homography method, it was decided to employ corner detection.

A corner can be defined as the intersection of two lines. These lines, in the case of the project, are the edges of the shapes lit on the array. A shape can be defined, or at least it could be guessed, from its corners. It can be seen more clear on figure 2.3, with only the corners, the shape can be guessed. A better identification happens when there are more points available, but that would make the whole process time consuming.

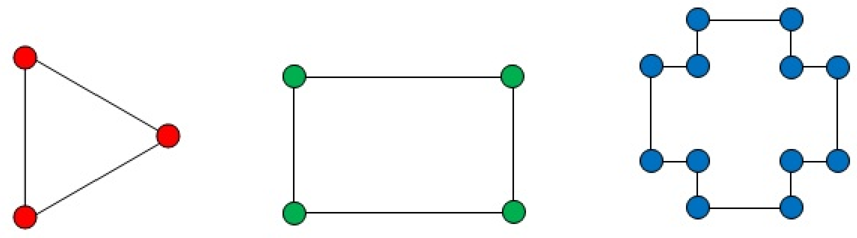

(a) Triangle (b) Rectangle (c) Cross

Figure 2.3: Corners on different shapes

These corner points can be used for the homography, that's why it is important to extract them correctly from the glint shape. Harris corner was employed because it has strong invariance to: rotation, scale, illumination variation and image noise.

In order to find the corners, Harris detector looks for areas of the image or windows that when shifted in any direction generate a large change in appearance. In Figure 2.4 it can be seen more clearly.

It is clear that on picture 2.4a if the red window is shifted around, there will be no large variations. On the other hand, on picture 2.4b it can be

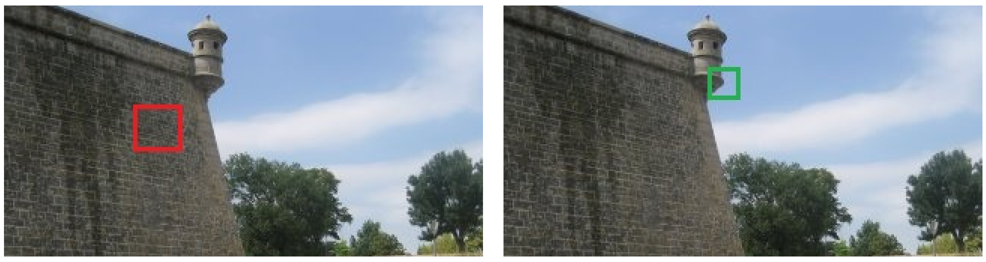

(a) Window without variation (b) Window with variaton

Figure 2.4: Example of two windows used for Harris Corner detection

seen that when shifting the green window, there will be large changes. Intuitively, it could be guessed that the area inside the red square will not have any corners: there are no big colour intensity changes. Following the same reasoning, under the green square, corners should be found: there are major colour intensity changes.

Harris corner detector can be translated into mathematics in the following equations:

$$E\left(u,v\right)=\sum_{x,y}w\left(x,y\right)\left[I\left(x+u,y+v\right)-I\left(x,y\right)\right]^2 \tag{2.3}$$

Studying Equation 2.3:

- $E(u,v)$: Represents the difference between the original and the displaced window. u is the displacement in the X-axis. v is the displacement in the Y-axis.
- $w(x,y)$: Window at the position (x,y).
- $I(x,y)$: Intensity at coordinates (x,y).

Expanding Equation 2.3 using Taylor series:

$$E\left(u,v\right)\approx\sum_{x,y}\left[I\left(x,y\right)+uI_x+vI_y-I\left(x,y\right)\right]^2 \tag{2.4}$$

$$E\left(u,v\right)\approx\sum_{x,y}\left[u^2I_x^2+2uvI_xI_y+v^2I_y^2\right] \tag{2.5}$$

Rewriting it as a matrix equation:

$$E\left(u,v\right) \approx \left[\begin{array}{cc} u & v \end{array}\right] \left(\sum \left[\begin{array}{cc} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{array}\right]\right) \left[\begin{array}{c} u \\ v \end{array}\right] \tag{2.6}$$

$$E\left(u,v\right) \approx \left[\begin{array}{cc} u & v \end{array}\right] M \left[\begin{array}{c} u \\ v \end{array}\right] \tag{2.7}$$

Where:

$$M = \sum_{x,y} w\left(x,y\right) \left[\begin{array}{cc} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{array}\right] \tag{2.8}$$

Computing the response of the detector:

$$R = Det(M) - k(Trace(M))^2 \tag{2.9}$$

This value R is calculated for every window. It is compared to another threshold value, and if R is greater than that threshold, then there is a corner.

2.4 RANSAC

RANSAC: "RANdom SAmple Consensus". It's an algorithm introduced in 1981 by Fischler and Bolles [4]. Its goal is to estimate the parameters of a certain model starting from a set of data contaminated by outliers. It produces a reasonable result only with a certain probability, that it's increased when more iterations are performed.

An outlier is a sample that doesn't fit the parameters of a certain model. Therefore, an inlier is a datum that does fit the parameters of a certain model. Figure 2.5 shows this:

RANSAC is applied to our case to minimize the mismatches that would negatively affect the homography. A set of points is detected, and then, using RANSAC, a more robust estimation can be done, finding out which points are inliers and which ones outliers. For deeper explanation on how RANSAC works, please refer to [6].

Once the points of interest have gone through RANSAC, the homography can be performed. The 4 correspoding points, extracted from all the inliers, are the optimal points for the homography.

The following section explains the second method employed in the project, Shape Context.

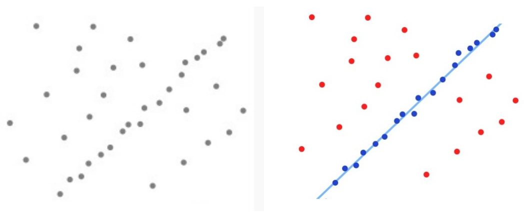

(a) Set of outliers (b) Fitted line

Figure 2.5: RANSAC applied to a set of points

2.5 Shape Context Principles

Another approach that is used in the project is Shape Context based matching. Shape context is mainly used for object recognition, as Belongie et al. [2] explain. They present a new way to measure similarity between two shapes, and use it for object recognition.

Basically, the process consists on two phases:

1. Find correspondences between the sets of points from the two shapes.
2. Based on the correspondences found, estimate an aligning transform.

Instead of working with feature points as in the case of homography (corners), this method uses edges. That's why the first step is to get the edges of the shapes analysed. Then, a certain number of samples is taken from the edges. These can be any type of points, they don't need to be feature points, just on the edge of the shape. So, for a point p_i from shape 1, a point q_i from shape 2 must be found as best matching point.

Belongie et al. [2] introduced a more robust descriptor: the distribution over relative positions. It consists on taking a point p_i on one shape, taking the coordinates of the remaning $n-1$ points, and calculate a histogram h_i of the relative coordinates:

$$h_i\left(k\right) = \left\{q \neq p_i : \left(q - p_i\right) \in bin\left(k\right)\right\} \tag{2.10}$$

That histogram h_i is the shape context for p_i. This process is repeated for every point. Then is time to calculate the cost C_{ij} of the matching. As mentioned before, h_i is the histogram for p_i and h_j for q_j.

$$C_{ij} = C\left(p_i, q_i\right) = \frac{1}{2}\sum_{k-1}^{K} \frac{\left[h_i(k) - h_j(k)\right]^2}{h_i(k) + h_j(k)} \tag{2.11}$$

That cost function needs to be minimized (equals to minimize the cost):

$$H(\pi) = \sum_i C\left(p_i, q_{\pi(i)}\right) \tag{2.12}$$

being π a permutation.

Once there is a set of correspondences between the points of the shapes, they can be mapped from one shape to the other. In this approach, the Thin Plate Spline (TPS) is used, as it is used for representing flexible coordinate transformations. Previously, all points p_i with coordinates (x_i, y_i) are unique and non-collinear. The TPS interpolant $f(x, y)$ minimizes the bending energy I_f as follows:

$$f(x, y) = a_1 + a_x x + a_y y + \sum_{i=1}^{n} w_i U\left(\| (x_i, y_i) - (x, y) \|\right) \tag{2.13}$$

To model the coordinate transformation, two TPS functions (see Equation 2.13) are used:

$$T(x, y) = (f_x(x, y), f_y(x, y)) \tag{2.14}$$

With the aid of Equation 2.14, any point from the first shape is mapped to its interpolated location on the second shape. Once the matching and TPS is done, what follows is an estimating of the shape distance: weighted sum of shape context distance, image appearance distance and bending energy.

Shape context distance

It is the symmetric sum of shape context matching costs from the best matching points extracted from shapes P and Q.

$$D_{sc}(P, Q) = \frac{1}{n} \sum_{p \in P} argmin_{q \in Q} C(p, T(q)) + \frac{1}{n} \sum_{q \in Q} argminp \in PC(p, T(q)) \tag{2.15}$$

Image appearance distance

Once correspondences are calculated, the distorted image can be transformed back into its "normal" shape: getting the points in one shape and moving them until they end in the coordinates of the points of the other shape. That process has a cost, appearance cost:

$$D_{ac}\left(P,Q\right)=\frac{1}{n}\sum_{i=1}^{n}\sum_{\Delta\in Z^2}G\left(\Delta\right)\left[I_P\left(p_i+\Delta\right)-I_Q\left(T\left(q_{\pi(i)}\right)\right)+\Delta\right]^2 \quad (2.16)$$

where:

- I_P: Gray-level image of shape P.
- I_Q: Gray-level image of shape Q.
- Δ: Differential vector offset.
- G: Windowing function, typically a Gaussian.

Benging energy

The most clear definition for "Bending Energy" is the amount of transformation necessary to align the shapes. For calculating it, first it is necessary to obtain the TPS interpolant $f(x,y)$, from Equation 2.13. This function minimizes the bending energy I_f:

$$I_f=\iint_{\Re^2}\left(\frac{\partial^2 f}{\partial x^2}\right)^2+2\left(\frac{\partial^2 f}{\partial x\partial y}\right)^2+\left(\frac{\partial^2 f}{\partial y^2}\right)^2 dxdy \quad (2.17)$$

Chapter 3

Experiment

This Chapets describes the process that started with choosing the components, programs, etc., went through designing the experiment, acquiring and processing data and ended with getting the results.

3.1 Design Parameters

3.1.1 Matlab

Matlab is a programming environment, similar to other programming languages such as C and C++. It works faster than them when dealing with technical problems, and it is widely used in the image processing field. This is one of the reasons for using Matlab, since this project works with frames taken from a camera.

It's been used for almost 30 years, so the amount of bibliography, code, forums and information in general is big. That is another reason for using Matlab. In fact, part of this project uses published code, adapted where needed.

It has the capability to work with different devices, sending and receiving data from them. A webcam and an Arduino board were going to be used, and Matlab works perfectly with both of them: It gets the frames or video feed from the camera; it sends and receives information to and from Arduino board.

3.1.2 Arduino

Matlab alone could not handle the LEDs employed in the project. It therefore became necessary to find a device or instrument capable of working with both Matlab and the LEDs. The Arduino board fulfils this task. From the different models of Arduino boards, the Arduino Uno was chosen

Arduino Uno has many digital and analog inputs and outputs (used for controlling the LEDs) and USB connection (for working with Matlab on the computer), among other features. It also has some memory space, where the programs are loaded once they've been compiled on the computer.

It is very versatile, it can work with LEDs alone or, as is the case in this project, with an LED driver. In this project it has been used only for sending data (mainly orders) to the LED driver, but it also has the option of receiving data from sensors.

3.1.3 LED-Driver

A basic element of this project was to test the idea of using more than 4 LEDs to achieve its objectives. Also, this device/controller needed to be able to work with either Matlab and/or the Arduino. The MAX7221 LED driver seemed to be the perfect fit: a lot of information available on forums showed how to make them work together.

Another reason to choose it is its ability to control more than 4 LEDs. More specifically, a single MAX7221 LED driver can work with a maximum of 64 individual LEDs. As can be read later, more than one LED driver can be connected on cascade and work with just one Arduino board.

3.1.4 Light Sources

As mentioned previously, the light sources used in the project are infrared LEDs. They meet the specifications of the LED driver, so they work perfectly. When working with eye/gaze tracking systems, it is usual to employ infra-red (IR) LEDs. They are active light sources.

3.1.5 Camera

This project used a regular webcam, a Sandberg NightCam. As the name suggests, it has night vision capability. This is a really interesting feature, meaning that it can detect infrared light (like the one the LEDs employed emit). A 12mm lens has been used with the camera in the project.

3.2 Settings

This section describes how the most relevant connections have been made. After that, it will show the different shapes used in the project.

3.2.1 Matlab - Arduino

The connection was made through an USB cable: from one USB port in the computer to the USB port in the Arduino board. A function was written in Matlab that enquired the user in order to choose the shape to be lit on the array. Matlab sent that information to the Arduino board, which performed according to the data received.

3.2.2 Arduino - LED driver

Figure 3.1 shows how to properly connect the Arduino board with the LED driver. The schematic can be found at [1, Arduino's webpage]. The code for controlling the LED driver was written with Arduino environment. It controlled the intensity light the LEDs emitted, as well as the exact LEDs that had to be turned on. Once the code was written, it was compiled and uploaded to the board.

The bottom of 3.1 contains the following text: ”to next MAX72XX”. As it has been mentioned before, this shows that more than one LED driver (like the one used in this project) can be used, if they are employed in a cascade configuration. As mentioned in [1], there is no maximum amount of LED drivers than can be cascaded; the limitation comes from the length of the cables between MAX7221 and also from the power sources available. So, ideally, the number controlled by a single Arduino board varies on multiples of 64 LEDs (using the maximun capacity of the LED driver).

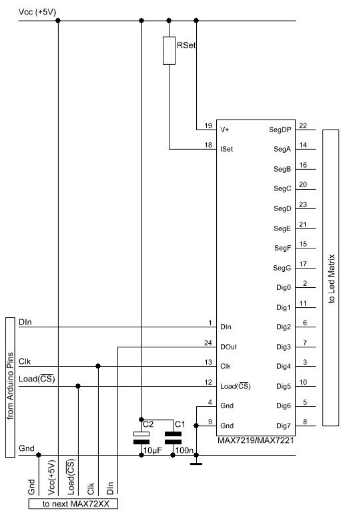


Figure 3.1: Schematic for connecting the Arduino board and the MAX7221

Figure 3.2 shows how the connections have been implemented on the project:

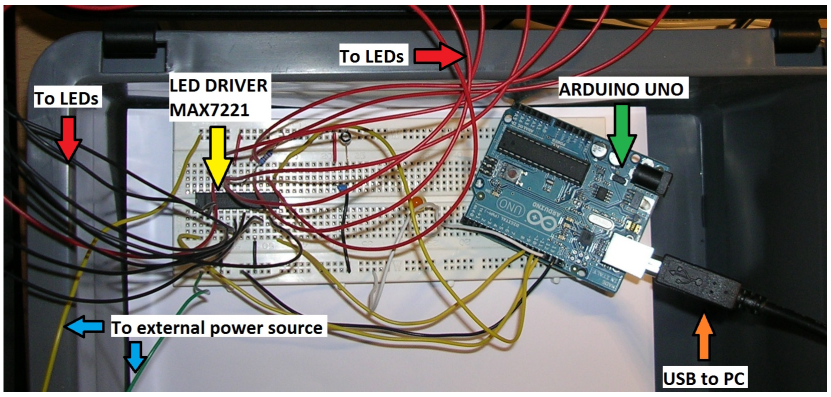


Figure 3.2: Connections between Arduino Uno and MAX7221

As can be seen in picture 3.2, the available space is the limit for adding more LED drivers. From each LED driver using its maximum capacity comes just 16 wires for linking with the LEDs. It would be necessary to add more for power supply, connection with Arduino, etc. It is clear, as written above, that a limit exists in the number of devices, coming from the length of the cables and also here from the space available.

3.2.3 LED driver - LEDs

For connecting the 64 LEDs to the LED driver there is a schematic 3.3 provided by [1, Arduino's webpage]. The labels (Dig0, SegDP and similar) show where to connect this points in the LED driver.

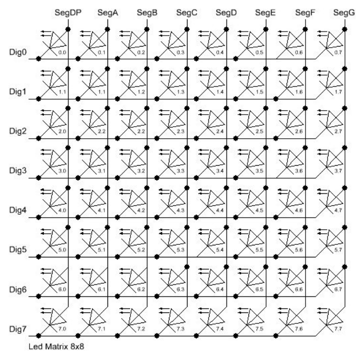


Figure 3.3: Schematic for connecting the LEDs to the MAX7221

This is how it actually looks 3.4:

3.2.4 Shapes

One of the foundations of this projects is shapes. Geometric shapes that are lit on the custom built LED array. The criteria used to choose them was:

- Recognizable shapes. Shapes that could be drawn on a 8x8 LEDs array.
- Different types of shapes. Having only variations of the same shape would not have any benefit to the project. It was better for them to be different so their performance could be tested.

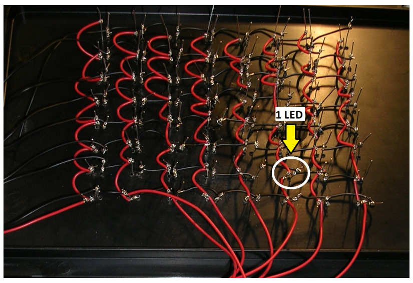
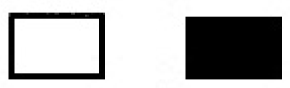


Figure 3.4: Capture of the connections between MAX7221 and the LEDs

- Enough quantity. Not too many or too little, just the right amount in order to perform a valid study.

Taking all these into consideration, it was decided to use 8 different shapes. Next step was to figure out what the shapes should look like: work only with the edge of the shape, or use the full shape. In figure 3.5 it is can be seen more clearly:



(a) Edge shape (b) Full Shape

Figure 3.5: Edge Shape vs. Full Shape

Early tests were carried out using "edge shapes", but since the results were not good enough, it was decided that only 2 out of 8 shapes would be of this kind. This way, the performances of these two types could be compared under the same working conditions.

Shapes on the array

Once all the connections between the devices had been made, it was time to start lighting on some LEDs in order to "draw" the shapes. The code was written in Arduino's own environment, which is based on C/C++, compiled with the programme and then uploaded to the board. The function used for this purpose it's "*shapes full.pde*".

The function is divided as it follows:

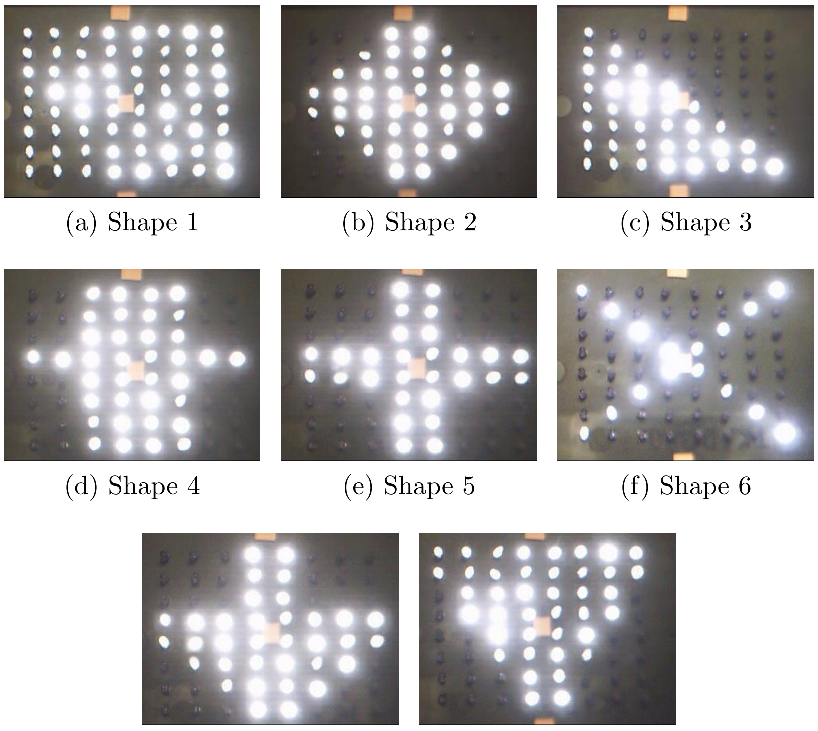
- Loading library "*LedControl.h*". It's a specific library created for controlling the MAX7221 from the Arduino board.
- Initialization:
  1. Initializing the variables used in the program.
  2. Opening the communications with Matlab through a serial connection
  3. Setting the initial state of the LEDs (OFF).

4. Setting the intensity the LEDs were going to have through the experiment. It can vary on a scale from 0 (minimum) to 15 (maximum).

- Drawing the shapes:

1. Receives the data coming from Matlab (one byte at a time).
2. Reads the data, more specifically, the variable "*matlabData*" and finds out which shape should be turned on.

The 8 shapes, as lit on the array, can be seen in figure 3.6.



(a) Shape 1 (b) Shape 2 (c) Shape 3

(d) Shape 4 (e) Shape 5 (f) Shape 6

(g) Shape 7 (h) Shape 8

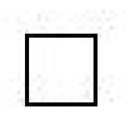
Figure 3.6: Images of the shapes used in the project. They are lit on the LED array.

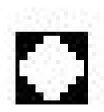
The two edge shapes employed for the project are Shape 5 (3.6e) and Shape 6 (3.6f). The difference with the rest of the shapes is clear: the amount of LEDs turned on, the intensity of the light the project, etc.

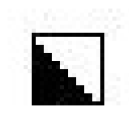
Ideal Shapes

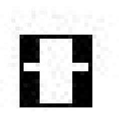
As will be explained later on the Analysis section 3.5, there is a need to compare the glint we extract from a frame with something in order to evaluate the performance of the different shapes and methods. For that, an ideal shape (as ideal as it was possible to make) data base was created from scratch.

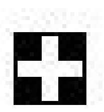
Each of the 8 shapes were replicated on a 16x16 pixel matrix, created with Matlab. They try to replicate the shapes lit on the LED: 1 LED from the array is translated into a 2x2 pixels subarray. An extra edge was added in order to help Matlab work better with the shapes. The results can be seen in figure 3.7

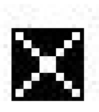

(a) Shape 1

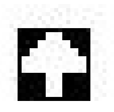

(b) Shape 2


(c) Shape 3


(d) Shape 4


(e) Shape 5


(f) Shape 6


(g) Shape 7

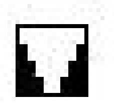

(h) Shape 8

Figure 3.7: Shapes from the "ideal" database created with Matlab

3.3 Scenarios

In order to evaluate the performance of the shapes, it was necessary to carry out the tests in different environments or scenarios. There was two main scenarios:

- Indoors
- Outdoors

As it will be seen in the Analysis section 3.5, more tests were done indoors than outdoors. It was not easy to take out all the equipment; also, the people that took the tests had limited time, so it was decided to perform more tests indoors in order to have more data to compare under the same circumstances.

The different scenarios in which the tests were done will be described next. Different tests had different scenarios.

3.3.1 Points of Analysis

In order to make a better anaylsis and comparison, every test was carried out with the user looking at the same 5 points, in the same order. They were points located straight in front of the person, and were numbered from 1 to 5. This way, every test was made under the same procedure. It is illustrated in picture 3.8.

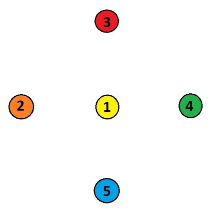


Figure 3.8: Points to look during the tests

3.3.2 Test 1

The goal of this test was to get samples from various scenarios from one single user. The data was processed and analysed as shown in section 3.5.

It was divided into 6 different scenarios. It was done like this in order to do a first test with as many positions (camera and array) as possible, narrowing afterwards the better conditions. There were two scenarios (A and B) with the camera mounted on top of the array, and four scenarios (W, X, Y and Z) with the camera mounted on a separate support arm. Table 3.1 shows the different positions for the camera and the array depending on its scenario.

Table 3.2 shows the distances at which the array and support arm were placed.

For this test, the user was sitting and had the array in front. Also, the points of analysis were right behind the array (straight line). The support

	Array		*Support Arm*	
Scenario	Position 1	Position 2	Position 1	Position 2
A	X		-	-
B		X	–	-
W	X			X
X		X		X
Y		X	X	
Z	X		X	

Table 3.1: Position in the scenarios

	Array		*Support Arm*	
Scenario	Position 1	Position 2	Position 1	Position 2
A, B	16cm	23cm		
W, X, Y, Z	16cm	23cm	17cm	26cm

Table 3.2: Distances for array and support arm

arm was standing on a side, with its end hovering over the array.

It was done by one subject.

3.3.3 Test 2

The goal of this test was to get samples from two different scenarios and various users. How they were processed will be analysed on section 3.5

This test had only two scenarios (B and C). The array with the LEDs was located on the right side of the user, pointing at him/her. It didn't change its location for this test. The camera was mounted on the support arm, which was standing as well on the right side (like in the previous test). The two scenarios mentioned previously (B and C) correspond to two different settings of the support arm, different distances to the edge of the table (and as a consequence, to the user). These distances are written in Table 3.3.

	Support Arm	
Scenario	Position 1	Position 2
B	12 cm	-
C	-	26 cm

Table 3.3: Distance from Support Arm to Edge on Test 2

The skin and eye colour for the 6 people that did the test are:

1. User 1: Pale skin and dark brown eyes.
2. User 2: Pale skin and brown eyes.
3. User 3: Pale skin and light brown eyes.
4. User 4: Dark skin and dark brown eyes.
5. User 5: Pale skin and green eyes.
6. User 6: Pale skin and light brown eyes.

There are not that many different skin colours but, on the other hand, every individual had a different eye morphology. Meaning that there were bigger, smaller, narrower, etc. eyes. Next Figure 3.9 shows pictures from the analysed eyes.

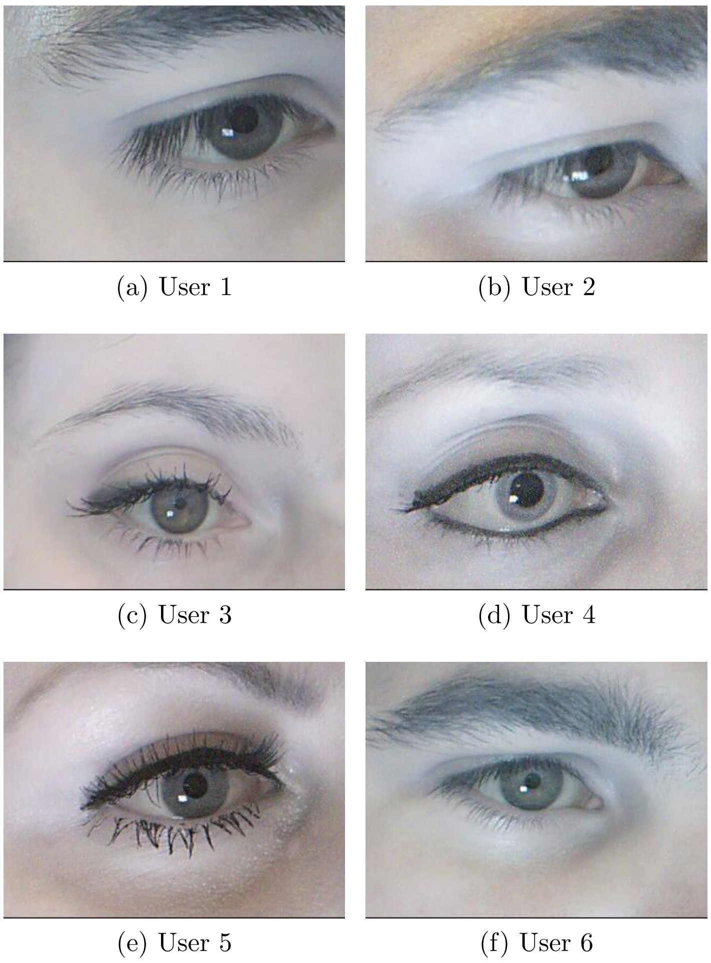
Each user had to follow the same procedure as in Test 1. The difference was that this time, each person only had two scenarios to work with due to time constraints.

3.3.4 Test 3

This test was an outdoor test. The goal was to see how the system performed in a more changeable environment than an indoor environment. There was one scenario and two different subjects. Data analysis can be seen on section 3.5

As mentioned above, Test 3 worked with one scenario (A): The camera was sitting on the support arm, which was placed 26cm away from the edge of the table. Two different users performed this test: User 1 and User 6 from Test 2, and their features can be read in 3.3.3.

This test was first conducted completely outdoors. More specifically the equipment was set at IT-University outdoor surroundings. The problem was that the camera couldn't get an image in those conditions. The image was almost completely white, and even dark eyebrows took a lot of effort to be distinguished (not to mention a small white glint). The intensity of the LEDs was changed as well, trying to improve the results, but no positive result was obtained. It can be seen in Figure 3.10



(a) User 1 (b) User 2

(c) User 3 (d) User 4

(e) User 5 (f) User 6

Figure 3.9: Eyes of the different users

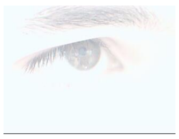


Figure 3.10: Outdoor shot of the eye

From this picture no data could be extracted, so it was decided to carry out the experiment in the IT-University lobby. Not the same conditions as on the surroundings outdoor, but still, not as steady and predictable as an office space.

Again, each user had to follow the procedure described at Test 1, but this time only for one scenario.

3.4 Data Acquisition

Early tests

Early tests were done in order to see how the code worked, if shapes could be recognised, changing parameters to improve the results. One interesting finding relates to the size of the glints on the eye. The minimum area in which a glint was not just a blur was 20 pixels, and the maximum was less than 120 pixels. All tests took this into consideration, adjusting slightly these values to narrow the search for the glint and avoid noise or bigger artefacts that had nothing to do with the glint.

Another interesting fact found on early tests relates to the location of the eyes within the image. Each image processed and analysed has a 240x320 pixels size. Analysing different pictures it was found that most of the times the center of the eye (and with it the glint) could be found within a smaller area. More specifically, 40 pixels from the top, bottom and both sides were "cut". This means that only centroids (this will be explained in the pre-processing section) that fell within that new area were taken into consideration for the processing. With this, shapes that could have interfered with the process were avoided.

Ideal Database

It's been mentioned on Section 3.2 the creation of a database with the ideal shapes. This was done in order to improve and speed up the calculations.

Using these ideal shapes as starting points, two more databases were created, one for each method analysed.

- Homography: Harris Corner detector was applied to the ideal shapes. As a result, a database was created, containing the coordinates for the feature points detected.

- Shape Context: Edge detector was applied to the ideal shapes. As a result, a database was created containing the coordinates for the points on the edges of the shapes.

Test Methodology

For every test, there was a set of steps that needed to be taken. The process was as follows:

1. Set the array and camera into the setting corresponding with the scenario being tested.
2. Choose a shape and light it on the array.
3. Sit the user on the right place.
4. Start looking at the analysis points and take a picture when the eye had stopped focusing on the point.
5. Move to the next point and repeat the process.
6. Once all 5 points were covered, the scenario was changed, and the process repeated.
7. Once all scenarios were covered for one shape, it was changed, and the process started from the top.

This was repeated until all 8 shapes were completely tested.

Following this process the amount of pictures available for processing and studying can be seen on Table 3.4.

	Users	Shapes	Scenarios	Pictures	*Total*
Test 1	1	8	6	5	240
Test 2	6	8	2	5	480
Test 3	2	8	1	5	80
				Total	**800**

Table 3.4: Number of pictures taken for the tests

3.4.1 Homography

There is a lot of bibliography and code that has been tested and could be used for the project. For example, Peter Kovesi has done an extensive work on Image Processing (among other things) using Matlab [7]. His functions have been used in this project, being modified in order to meet the needs of this project. The original work was extracted from Peter's functions webpage.

This process can be distinguished into two main parts: pre-processing the image and obtaining the homography. A final part is written in order to save the results. They are all explained in the following lines.

Pre-processing the image

Mainly, this part's objective is to make the image ready for Peter's functions. It is done like this:

1. Load the image/frame.

2. Convert it to a grayscale intensity image.

3. Create morphological structuring element. Dilate those morphological structuring elements. These two combined actions recover some of the shapes that have been lost when converting the image into a grayscale image.

4. Obtain the threshold for the image. Use this value to convert the grayscale image into a binary one.

5. Erase small dots of noise that could interfere with the process. All elements that have an area less than 20-30 pixels are eliminated from the image. The result, is a clearer picture.

6. Find connected components. From them, extract the ones that occupied more than 30 pixels area and less than 120. This portions are the candidates to being the searched glint.

7. From there, their centroid (center of mass) is found, as it will be needed in the following steps. As has been mentioned some lines above, only the centroids falling inside a specific area are taken into consideration.

Once these steps are done, the next step is to calculate the homography for the glint. Only if there is any shape found. If not, no calculation is done and the process skips to the next picture.

Obtaining the Homography

As has been said, for this part Peter's functions [7] are used, with some modifications. The steps that follow are the ones that take place for getting the homography for a single picture (if a glint has been found in the pre-processing phase).

1. Apply Harris' Corner Detector. If succesful, it returns the binary image marking corners, with row and column coordinates too.
2. Load the corners from the ideal shape. They are going to be compared with the ones obtained from the image.
3. Centre the coordinates from the ideal database with the ones from the image.
4. Run "*matchbycorrelation.m*". It matches feature points (from image and ideal shape) by correlation. If succesful, it returns the set of points that are a match. If not, it jumps to the next image.
5. Next step is calculating the homography. For entering this part, at least 4 points must have come out from "*matchbycorrelation*".
6. Homography is calculated by function "*ransacfithomography.m*". As it says, it fits a 2D homography using RANSAC. Among other things, it returns a 3x3 homography matrix (x2=H*x1).
7. Save the data.
8. Back to the beginning (if there are more photos left to process).

Saving the results

This process generates different types of data. There is some data that comes with every frame, and some data that comes from every scenario from a shape analysed.

Data from a frame:

- There is a Homography matrix,"*Hmg total*". If there were not enough points (or no points at all) detected, no homography matrix could be calculated. Therefore it was decided to write a 3x3 matrix filled with zeros. That way, they could be distinguished from the real homographies.

- There is a value "*total*" that acts as a flag: it is 0 there is no homography available for that particular image; if it is 1 there is a homography available.

Data from a shape:

- There is a value called "*percentage*". It shows, as the name itself says, the percentage of homographies successfully produced against the total amount of pictures analysed. Again, this is a value that comes for a single shape and all the scenarios tested (Test 1 has 6, Test 2 has 2 and Test 3 has 1 per shape).

Figure 3.11 shows the process in pictures.

3.4.2 Shape Context

As has been mentioned before (see Section 2.5), the Shape Context part of this project has been heavily influenced by the work carried out by Belongie et al., mainly from [2]. The authors also created some sample code, that can be found at this link. That example code has been used and modified for the necessities of the project.

As in the case of the Homography, there are two main parts (Pre-processing and obtaining the Shape Context). After that, a final part is written for saving the results.

Pre-processing the image

The goal of this part is to get the image ready for getting the Shape Context from each frame. The actions are the same made on the pre-processing for the homography, and can be seen in 3.4.1.

Obtaining the Shape Context

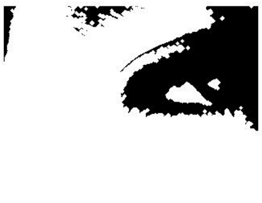
For this part of the project, the example functions from [2, Belongie et al.] have been used and modified to meet the specifications of the project. The process used is as it follows:

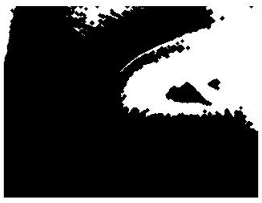
1. From the pre-processing there is an area of pixels that need to be analysed. More specifically, the centroid of that area.

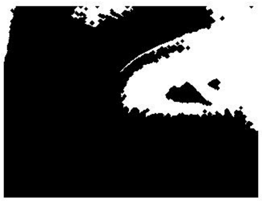
(a) Step 1

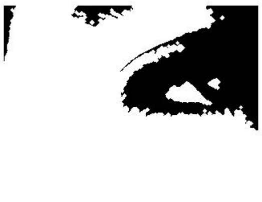
(b) Step 2

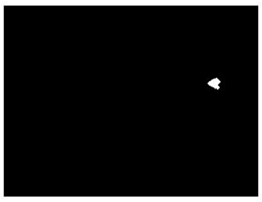
(c) Step 3

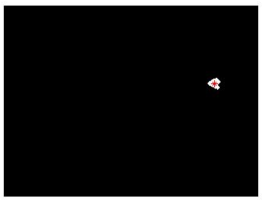

(d) Step 4

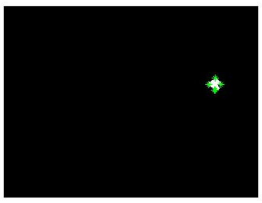

(e) Step 5

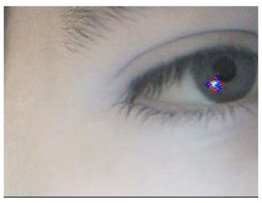

(f) Step 6


(g) Step 7


(h) Step 8


(i) Step 9


(j) Step 10


(k) Step 11

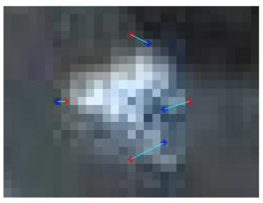

(l) Step 11 - zoom

Figure 3.11: Pictures from the Homography method.

2. From the image that has been processed (in this stage it is on a binary scale), an area of 20x20 pixels is cropped. In that area lays the glint that's going to be analysed.

3. The coordinates of the points of that shape are taken out from the cropped image and saved into a separate array.

4. One of Belangie's functions starts working. This is the one that's going to generate a matching based on Shape Context.

5. This function gets the coordinates of the shape from the image. Also, it loads the coordinates of the ideal shape from the specific database.

6. The matrices with the coordinates get re-shaped, as they both need to have the same size for Belangie's function.

7. Next step is to actually do the matching.

8. Once the process is done, results are saved and the process starts again with the next picture.

Saving the results

In this case, we obtain data from each frame. Amongst this data, two pieces are of particular interest for the purposes of this project (both are mentioned in section 2.5).

- Error. Is the difference between the shapes.

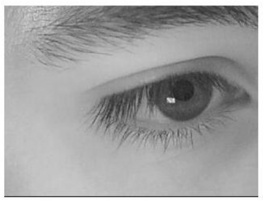
- I_f. This is also known as bending energy. It's the amount of energy necessary to align the ideal shape and the glint shape

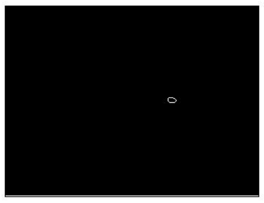
Figure 3.12 shows the process in pictures.

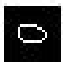
3.5 Analysis

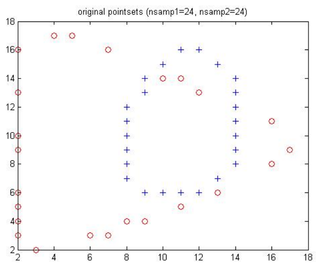
This section's goal is to analyse all the data extracted from the different tests carried out. There are going to be two main sections, as the data is going to be analysed with Homography and Shape Context.

(a) Step 1

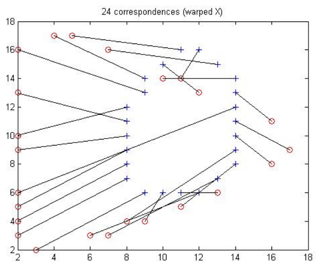

(b) Step 2


(c) Step 3

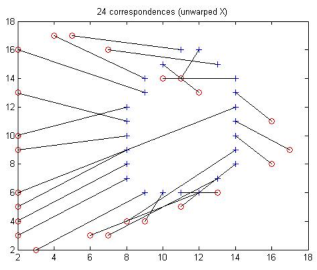

(d) Step 4



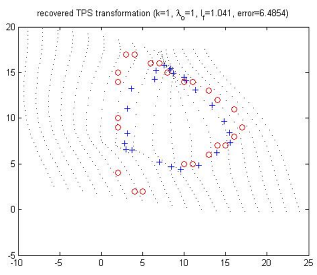
(e) Step 5



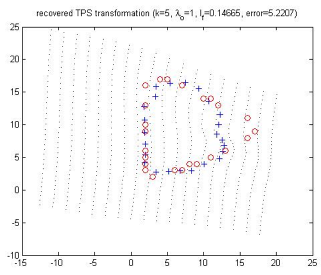
(f) Step 6



(g) Step 7



(h) Step 8



(i) Step 9

Figure 3.12: Pictures from the Shape Context method.

3.5.1 Homography

From all the data that came from the Homography process, "*percentage*" is the one this part is going to focus on. It shows the percentage of homographies successfully calculated. In other words, it shows the amount of shapes successfully detected. As a result, this value is an indicator of how good or bad a shape works under different circumstances (scenarios in this project).

As will be seen in the following tables, not all the shapes are analysed. Shapes 3, 5 and 6 do not appear in this analysis for the following reasons:

1. *Shape 3*: Bringing back the "ideal shapes" (Fig 3.7), it can be seen that number three is a right triangle. When extracting its feature points for applying Harris, more than three were collinear. That particularity made this shape undesirable for the test. In fact, when running Matlab's program, it kept crashing because of this. It could have been manipulated in order to make the analysis possible (moving the points so they were not collinear). But it was discarded for two reasons:

    (a) It would have implied changing the ideal shape, and therefore, the results would not have been right.

    (b) The process of getting the feature points was the same for the 8 shapes. Since a comparison between all of them was going to be made, the 8 must be processed the same way in, equal circumstances.

2. *Shapes 5 and 6*: The code from Peter's Functions [7] could not process these two shapes. Having the smallest amount of pixels lit , the glint produced was too thin for processing, having the consequence of not being able to produce any result.

It was mentioned at Section 2.2, that two values could be varied when calling for "*matchbycorrelation.m*": dmax and w. It was decided to run the code varying these two values, with the goal of discovering which conditions the shapes worked best under. For each value of w, all the values of dmax were simulated. These values are written in Table 3.5:

It makes a total of 600 different simulations for each shape and user.

	Initial	Final	Step	Total
w	31	91	2	30
dmax	13	53	2	20

Table 3.5: Values for varying ”w” and ”dmax” parameters

Test 1

As mentioned before, this test was done by 1 user on 6 different scenarios. What is represented in Table 3.6 is the percentage of Homographies calculated from the total of pictures taken. It is the mean of all the simulations done for each shape.

	Shape 1	Shape 2	Shape 4	Shape 7	Shape 8
Percentage	17.78	76.67	51.11	32.22	53.33

Table 3.6: Test 1 - Homography Percentages

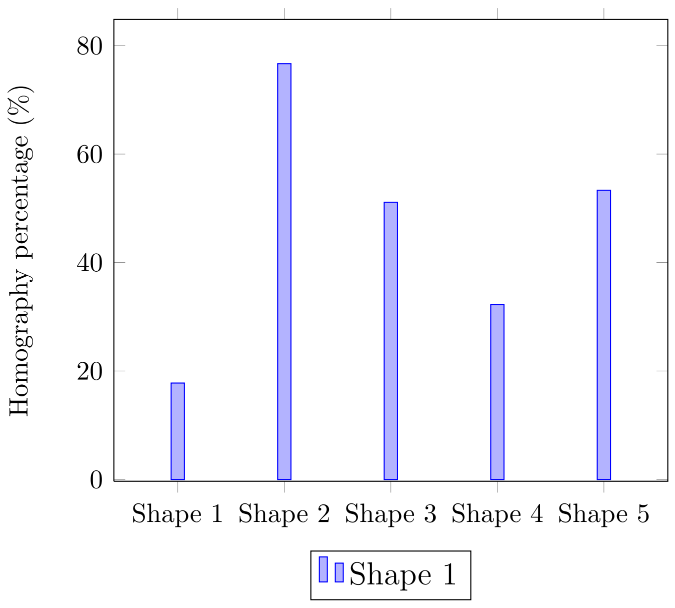


Figure 3.13: Test 1 - Homography Percentages

Shape 2, with 76.67% of homographies in total, is the one that works the best. Also, Shape 4 and Shape 7 work reasonably well, with percentages of 51.11% and 53.33%. As Shape 1 has the highest number of LEDs turned on, it may be assumed that it would have produced the largest glint on the eye and therefore performed well. The results show however, that it was the lowest with 17.78%.

Test 2

On this occasion, 6 different people did the test on 2 scenarios. For each shape, the mean of the percentages is calculated. Results are presented in Table 3.7.

	Shape 1	Shape 2	Shape4	Shape 7	Shape 8
User 1	2.22	15.56	14.44	31.11	15.00
User 2	3.33	13.33	14.44	15.56	17.78
User 3	1.11	16.11	5.56	17.78	11.67
User 4	3.33	7.78	8.89	15.56	10.00
User 5	9.14	5.45	0.56	21.62	13.94
User 6	10.00	17.78	7.78	15.56	15.56
Mean	*5.78*	*13.78*	*8.89*	*19.9*	*13.99*

Table 3.7: Test 2 - Homography Percentages

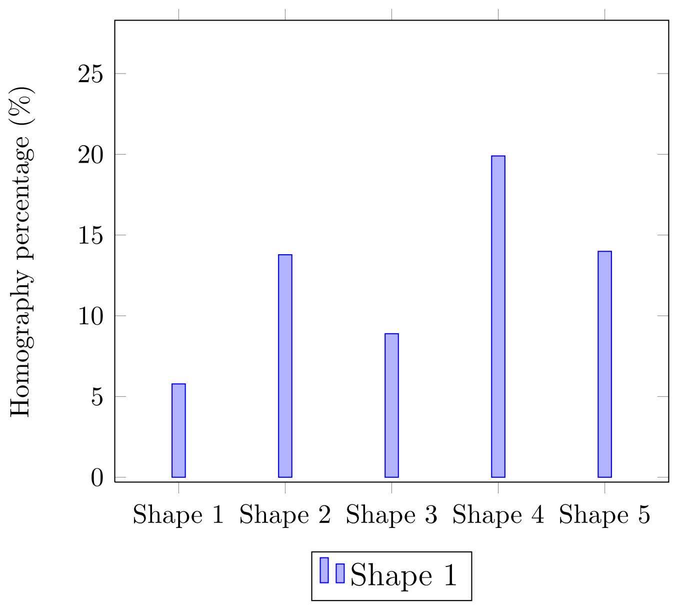


Figure 3.14: Test 2 - Homography Percentages (mean value)

As can be seen on Table 3.7, the shape that has the best performance is Shape 7, followed by Shape 8 and Shape 2. This data confirms the results extracted from Test 1 (see Table 3.6), although in different order. It also confirms the fact that Shape 1 has the worst performance of them all.

Another interesting analysis that can be done (using Table 3.6 and Figure 3.15) is studying how the system works with a certain type of person, focusing on his/her skin and eye colour:

- Pale Skin. There are no big differences in general between users with pale skin. 4 out of 5 users show, on average, similar values, with User 5 producing the lowest results. The performance of the shapes vary, of course, but the differences are reasonable (they have the same tendency).
- Dark Skin. User 4's performance is similar to pale skin user's performance. Its percentages are lower than the rest, but the differences between shapes on User 5 keep the margins the others also have.
- Brown eyes. It is interesting to compare the different shades of brown:
  - Dark brown. Even if the values keep proportions between themselves, User 1 has more homographies than User 4.
  - Medium brown. User 2. Its performance is close to the average performance of the rest of the users.
  - Light brown. Users 3 and 6. Both have similar performance, except when working with shape 1. The rest are very similar and also they are close to the average.

  If a comparison between all users with brown eyes is made, they all have similar performances, with little exceptions. The differences are not originated by the colour of the eyes.
- Green eyes. User 5 has a similar performance compared to the others, with the exception of Shape 4, which has the lowest percentage of homographies of all the tests.

Comparing all of them together, it can be seen that the shape that doesn't have values too far from each other and has a higher rate of homographies is Shape 8. Shape 7 has higher percentages, but it is not too stable. Shape 1 in general offers the poorest results, regardless of skin or eye colour.

Continuing with the comparison based on skin and eye colour, it is interesting to have a look at the average behaviour in general (as shown in Table 3.8). Their results are close to each other, not varying too much, with the biggest difference around 6%.

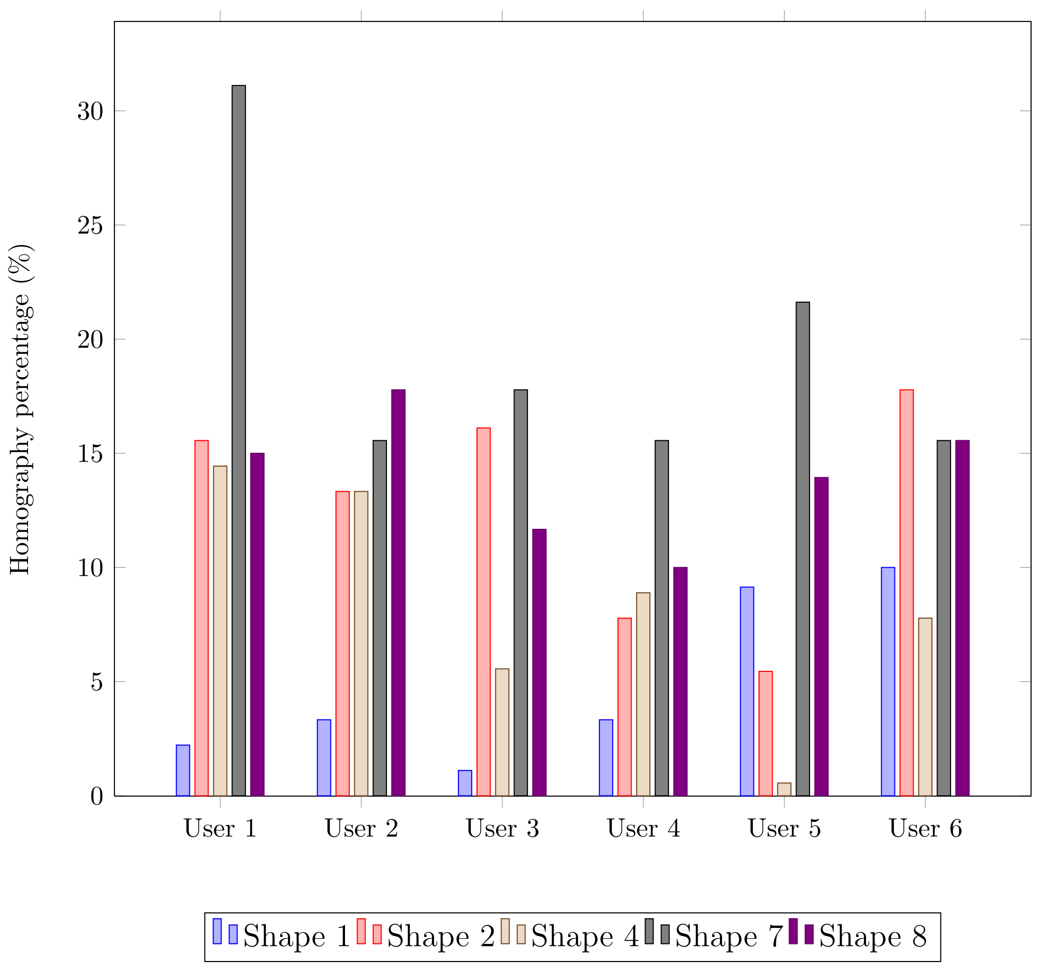


Figure 3.15: Test 2 - Performance of the shapes on different users

	User 1	User 2	User 3	User 4	User 5	User 6
Total	15.67	12.89	10.44	9.11	10.14	13.33

Table 3.8: Test 2 - Mean of percentages for each user

Test 3

As has been mentioned before, this test was done by User 1 and User 6 outdoors, working in 1 scenario.

Studying Table 3.9 the amount of homographies performed is very low, with no Shape or user getting more than 6%. User 6's performance is practically null, with Shape 2 being the only non-zero performer (but with a 1.1% rate of success).

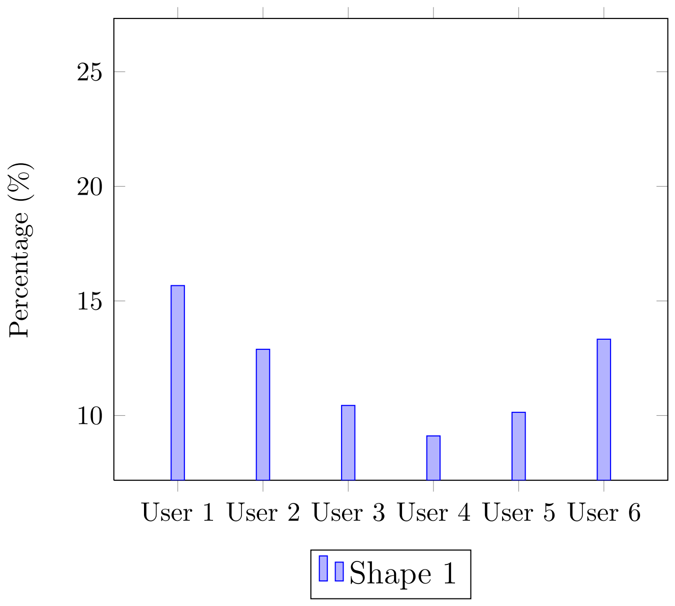


Figure 3.16: Test 2 - Mean percentage per user

	Shape 1	Shape 2	Shape4	Shape 7	Shape 8
User 1	5.56	5.56	2.22	2.22	0
User 6	0	1.11	0	0	0

Table 3.9: Test 3 - Percentages

If the pictures taken for Test 3 are analysed, it can be seen that their quality is much lower than the quality from indoor pictures. This means that if a better camera had been used, the results for these two users would have been expected to be similar to the results from Test 2.

Looking at Table 3.10, no decision can be made about which shape works better in outdoor environments. The data used for this analysis is not of good quality.

	Shape 1	Shape 2	Shape4	Shape 7	Shape 8
Total	2.78	3.33	1.11	1.11	0

Table 3.10: Test 3 - Mean of percentages for each shape

3.5.2 Shape Context

To analyse the performance of this part of the project (Shape Context) there are two main indicators: "Error" and "Bending Energy". This section will assess which of these two indicators is more reliable when matching shapes, which shape works better with each indicator, etc.

Also, it is worth mentioning how the tests have been designed. In this case, as opposed to Homography analysis, every single shape from the glint has been compared to the 8 ideal shapes from the database. This way, the Shapes have been tested, seeing if they actually match or not the one extracted from the glint.

For Shape Context analysis, all shapes have been tested as opposed to Homography's, where Shapes 3, 5 and 6 were discarded on previous stages.

Test 1

Let's explain what is represented in these two tables: The rows show "Ideal X" and the columns "Shape Y". It means that "Shape Y" was lit on the array and when running the code it was compared with the "Ideal Shape X". [1]

Tables representing Ideal X vs. Shape Y should be read like this:

- "Ideal X" is the ideal shape that it's loaded in Matlab.
- "Shape Y" is the shape that was lit on the LED array.
- The element that sits on the cell correspindg to Ideal X - Shape Y is:
  - If analysing the error: it's the difference between the shape extracted from the glint (created by Shape Y) and the ideal shape loaded in Matlab (that comes from Ideal X).
  - If analysing bending energy: represents the amount of energy necessary to align the ideal and the glint shape.

Table 3.11 shows the amount of error calculated. The element in the diagonal should be the lowest or one of the lowest in the row and in the column. That would show that the difference between the ideal shape and the glint it's the lowest, so it would be a match.

[1]In the following Tables, "Shape" will be written "Shp" due to space constraints

	Ideal 1	Ideal 2	Ideal3	Ideal 4	Ideal 5	Ideal 6	Ideal 7	Ideal 8
Shp 1	5,77	4,68	5,06	4,73	4,47	4,93	4,52	4,82
Shp 2	5,65	4,54	5,24	4,67	4,54	4,78	4,55	5,34
Shp 3	6,31	5,06	5,14	4,69	4,33	5,23	4,86	5,08
Shp 4	6,06	4,84	5,13	4,57	4,22	5,22	4,69	4,88
Shp 5	6,14	4,70	5,11	4,48	4,14	5,14	4,60	5,01
Shp 6	6,21	5,14	4,94	4,70	4,40	5,40	4,90	5,22
Shp 7	6,14	4,90	4,98	4,79	4,22	5,14	4,63	5,04
Shp 8	5,70	4,17	3,90	4,22	3,33	4,73	4,89	4,28

Table 3.11: Test 1 - Error Percentage

Analysing Table 3.11 it can be seen again that Ideal Shape 1 doesn't perform that good. In fact, it has one of the biggest error of the whole Table. That should be reason enough to discard using it. Following that reasoning, Shape 3 should be discarded as well (it would confirm what was said when analysing Homography). Other rejected Shapes were 5 and 6: Shape 6 has a 5.40% of error, being disqualified (again, confirming the supposition). That wouldn't be the case of Shape, that with 4.40% is way far from those values. The rest of the Shapes have good performance, having low values on the diagonal or close to the minimum of their respective row and column. Since it's not too much difference, they are accepted.

Bending energy could be, at first, a good indicator of how different the glint is from the ideal shape. Again, as with the error of Table 3.11, it is expected that the values on the diagonal are the smallest on their respectives rows and columns. This would indicate that the glint is very similar to the ideal shape, and therefore, a true match. Table 3.12 shows the bending energy values for Test 1.

	Ideal 1	Ideal 2	Ideal3	Ideal 4	Ideal 5	Ideal 6	Ideal 7	Ideal 8
Shape 1	0,20	0,10	0,21	0,17	0,14	0,29	0,13	0,17
Shape 2	0,20	0,09	0,20	0,16	0,13	0,30	0,11	0,19
Shape 3	0,43	0,26	0,36	0,31	0,21	0,31	0,28	0,33
Shape 4	0,30	0,15	0,27	0,21	0,17	0,29	0,16	0,27
Shape 5	0,30	0,14	0,22	0,18	0,12	0,30	0,15	0,26
Shape 6	0,52	0,34	0,47	0,35	0,30	0,37	0,28	0,43
Shape 7	0,26	0,17	0,19	0,18	0,17	0,28	0,15	0,25
Shape 8	0,14	0,10	0,13	0,12	0,13	0,32	0,11	0,09

Table 3.12: Test 1 - Bending Energy

Shape 3 and 6 have the highest values of all. That means they are not suitable for this matching method. Shape 2 works well, its value when matching Shape 2 with Ideal Shape 2 is the lowest. For Shape 4, all values (not counting 3 and 6) are pretty similar, so that wouldn't be a good Shape to use. Same thing happens with Shape 7, they are around the same values. The one Shape that has a good performance, is Shape 8. Its bending energy is very low (when comparing Ideal Shape 8 with glint Shape 8) compared to the rest.

One thing that can happen is that a Shape has a low error percentage, but a high bending energy value, or the other way around. That means the matching is not correct. As such, in relation to finding a good match, it should be stated that both the error percentage and bending energy have low values. That would be a robust way of matching. Following this reasoning, Shape 8 is the one that has the best performance, and Shape 1 has the worst. Shape 2 also performs well, as does Shape 7. Adding to the discarded ones (3 and 6) it's Shape 5.

Test 2

Test 2 was carried out by 6 users on 2 scenarios. Even though they were used in the Test, Shapes 3, 5 and 6 will not be mentioned in this part of the analysis. Nevertheless, their values will be shown in the following tables for comparison purposes. Seeing them on Table 3.13 and Table 3.14 gives confirmation that they aren't appropriate for the analysis.

The variety of users who performed the experiment gives us the chance to look into how well or bad Shape Context matching works for different users. Their performance will be studied focusing on the error and bending energy, first separately and after that together. It was stated in Test 1 that the combination of both parameters is a more robust estimation.

It is interesting to see how the shapes work with Shape Context matching on the users, who have different skin and eye colours. This analysis will be done with the data extracted from Table 3.13 and Figure 3.17

- Pale Skin. All pale skin users have very similar performances, with User 5 the best one and User 6 the worst one.
- Dark Skin. User 4's results are not very different from pale skin users.
- Brown eyes. It is interesting to compare the different shades of brown:

	User 1	User 2	User 3	User 4	User 5	User 6
Ideal 1	6,78	5,50	7,65	6,58	4,83	8,44
Ideal 2	5,22	4,79	5,50	4,62	3,00	6,57
Ideal 3	5,17	4,51	5,26	5,00	3,58	6,43
Ideal 4	4,79	4,05	5,70	4,53	3,10	6,25
Ideal 5	4,62	3,69	4,65	4,36	2,89	5,56
Ideal 6	5,10	4,44	5,62	5,09	3,55	5,89
Ideal 7	4,63	4,22	5,27	4,53	2,74	6,24
Ideal 8	5,00	4,42	5,51	5,15	3,25	6,61

Table 3.13: Test 2 - Medium error (percentage) per user

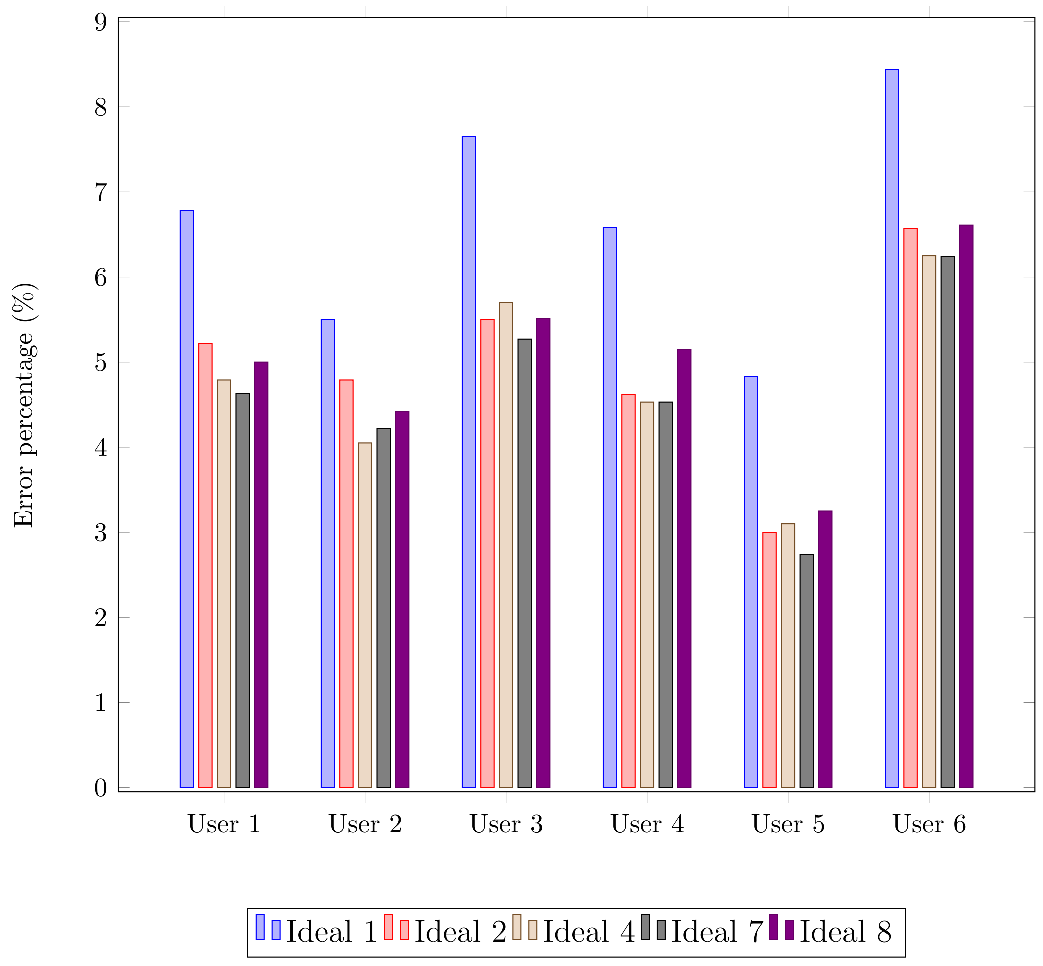


Figure 3.17: Test 2 - Performance of the shapes on different users - Error Percentage (%)

- Dark brown. There are no substantial differences between Users 1 and 4. Shape 7 is the one that works best for both of them, followed by Shape 4. Shape 1 offers the worst results.
- Medium brown. The results for User 2 are slightly better than the rest. In this case, Shape 4 is the best one, followed by Shape 7, but the difference is not very big. Again, Shape 1 is the worst.
- Light brown. Here the values differ between each other, being the ones from User 6 the worst of all Test 2. User 3 results are on the same level as the others.

If the results of all users with brown eyes are compared, they all fall around the same values, with the exception of User 6, that are much worse than the rest. In almost all Shapes, the error percentage is higher than 6%, and when using Shape 1 it goes up to 8.44%.

- Green eyes. User 5 has the best results in Test 2, when studying error percentage. It has the same tendency as the rest, but with lower values. Shape 1 offers again the highest percentages.

Skin colour differences don't affect the performance of the test, as can be seen. All shapes and user results move around the same values, meaning that eye colour does not make a difference in this test. The difference may originate in other factors like for example the morphology of the eye.

Now the same analysis is going to be made, but this time for bending energy. It will then be seen if these results coincide with the ones from error percentage or not.

	User 1	User 2	User 3	User 4	User 5	User 6
Ideal 1	0,51	0,18	0,54	0,49	0,15	2,89
Ideal 2	0,29	0,12	0,43	0,07	0,08	0,80
Ideal 3	0,35	0,18	0,33	0,23	0,14	0,80
Ideal 4	0,36	0,15	0,38	0,10	0,08	0,82
Ideal 5	0,28	0,12	0,32	0,09	0,07	0,68
Ideal 6	0,34	0,16	0,35	0,21	0,15	0,60
Ideal 7	0,32	0,12	0,43	0,11	0,09	1,06
Ideal 8	0,40	0,19	0,48	0,42	0,11	1,18

Table 3.14: Test 2 - Medium bending energy per user

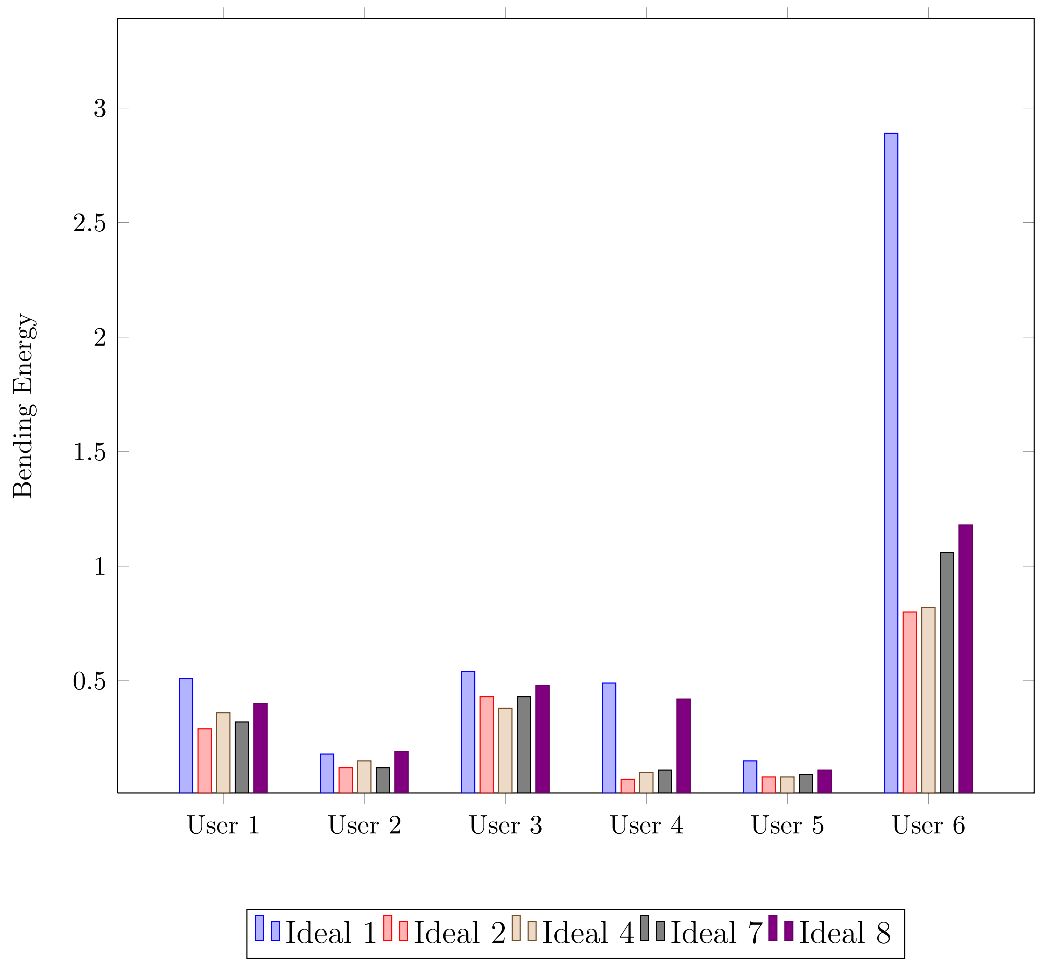


Figure 3.18: Test 2 - Performance of the shapes on different users - Bending Energy

It is interesting to see how the shapes work with Shape Context matching on the users, who have different skin and eye colours. This analysis will be done with the data extracted from Table 3.14 and Figure 3.18

- Pale Skin. User 6 is the one with the worst results. While Users 1, 2, 3 and 5 bending energy values aren't higher than 0.5, none of the values from User 6 are lower than 0.8. In pale skin users, the Shape that works the worst is Shape 1, but in this case the differences with the other shapes is small.

- Dark Skin. Shapes 2, 4 and 7 have low bending energy values. In fact, the lowest of the test. On the other hand, Shapes 1 and 8 are much higher, with differences of 0.3.

- Brown eyes.
  - Dark brown. Shapes 1 and 8 have almost the same bending energy values, but 2, 4 and 7 are much lower (around 0.10-0.15).
  - Medium brown. User 2 offers lower values, all the Shapes have similar performances, with differences no bigger than 0.07 between performances.
  - Light brown. On average, Users 3 and 6 have higher values than the rest. User 6's performance is, again, the worst of all.

  There are no big differences between users with brown eyes, if we exclude User 6. That difference in the values could be originated by bad quality pictures, different eye morphology, etc.

- Green eyes. The user with green eyes, number 5, shows the best performance of the test

These results confirm what was seen on error percentage: neither skin colour nor eye colour suppose a big difference between bending energy values. The Shape that works the worst again is Shape 1, but this time, its performance is closer to the rest.

Now it's time to analyse the performance of the shapes. It is going to be analysed if the shapes lit on the array (”Shp” in Tables 3.15 and 3.16, Figures) correspond, or not, to the Ideal Shape with it was compared. Shapes 3, 5 and 6 won't be analysed, as they were discarded before for their bad performance (their data will still be showed).

First it is analysed the average error per shape. The value in the diagonal (if there is Ideal X and Shape Y, where X=Y) should be either the smallest in its row and column or be close to that.

Seeing both Table 3.15 and Figure 3.19, it is clear that Shape 1 doesn't work well. It has the highest values among the other shapes, around 2% more. The rest of the shapes have very similar performances, no shape outshines the other. Shape 7 has the lowest error percentages and the difference between them is less than 1%.

Now is the turn of bending energy to be studied. As in the previous analysis, the value in the diagonal should be the smallest or if not, be close to that value.

	Shp 1	Shp 2	Shp 3	Shp 4	Shp 5	Shp 6	Shp 7	Shp 8
Ideal 1	5,49	6,50	7,12	7,20	6,80	6,65	6,51	6,76
Ideal 2	4,20	4,93	5,36	5,19	5,14	4,73	5,02	5,04
Ideal 3	4,38	5,18	5,49	5,16	5,12	4,72	4,78	5,10
Ideal 4	3,95	4,71	4,96	4,90	4,87	4,75	4,83	4,92
Ideal 5	3,79	4,31	4,66	4,58	4,34	4,26	4,21	4,22
Ideal 6	4,34	5,07	5,17	5,11	5,10	4,80	4,95	5,05
Ideal 7	4,13	4,63	4,77	4,84	4,80	4,58	4,59	4,51
Ideal 8	4,15	5,03	5,39	5,29	5,30	4,88	4,86	5,01

Table 3.15: Test 2 - Mean error per shape

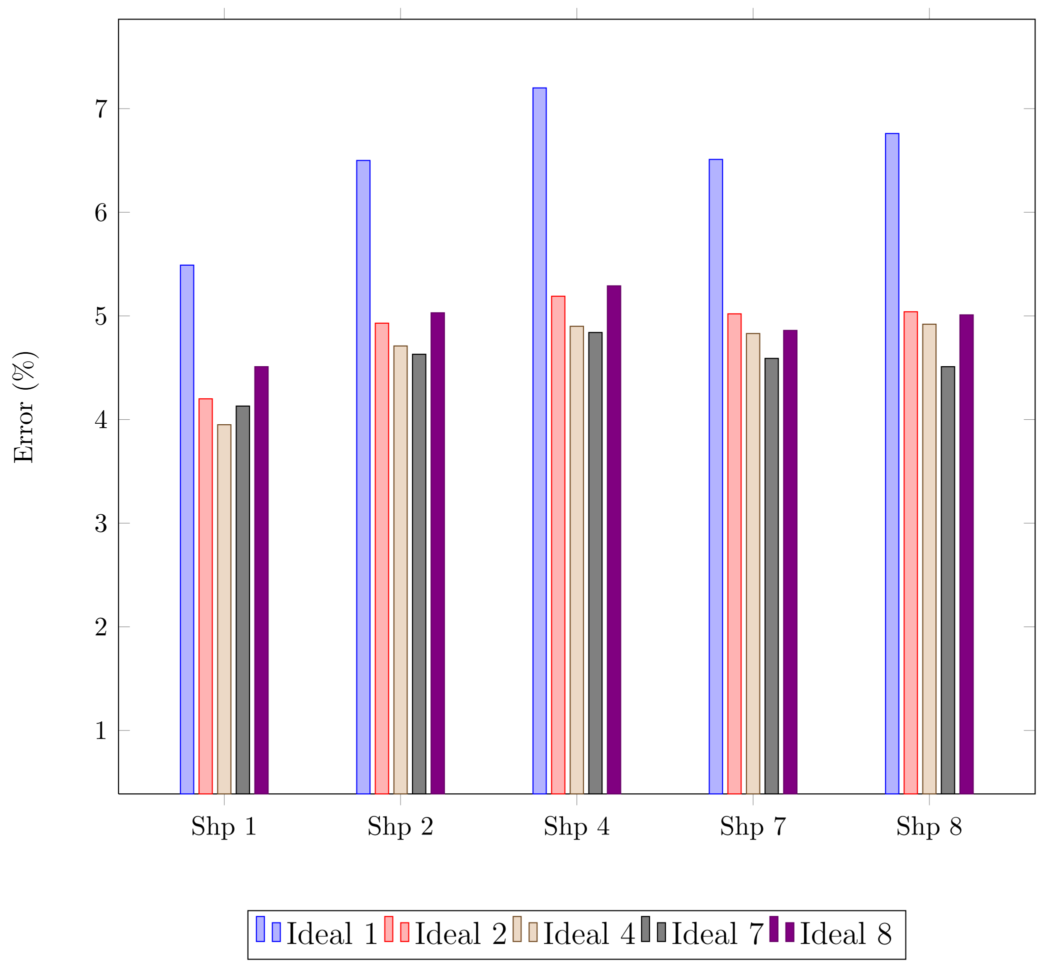


Figure 3.19: Test 2 - Mean error (%) per shape

	Shp 1	Shp 2	Shp 3	Shp 4	Shp 5	Shp 6	Shp 7	Shp 8
Ideal 1	0,85	0,84	0,81	0,96	0,70	0,78	0,68	0,72
Ideal 2	0,34	0,25	0,35	0,24	0,28	0,29	0,26	0,39
Ideal 3	0,34	0,30	0,41	0,33	0,38	0,39	0,30	0,27
Ideal 4	0,34	0,30	0,37	0,33	0,24	0,20	0,38	0,35
Ideal 5	0,22	0,24	0,29	0,24	0,29	0,28	0,26	0,25
Ideal 6	0,29	0,26	0,26	0,28	0,34	0,33	0,32	0,36
Ideal 7	0,32	0,32	0,35	0,44	0,46	0,46	0,27	0,23
Ideal 8	0,32	0,43	0,49	0,47	0,60	0,67	0,40	0,32

Table 3.16: Test 2 - Mean bending energy per shape

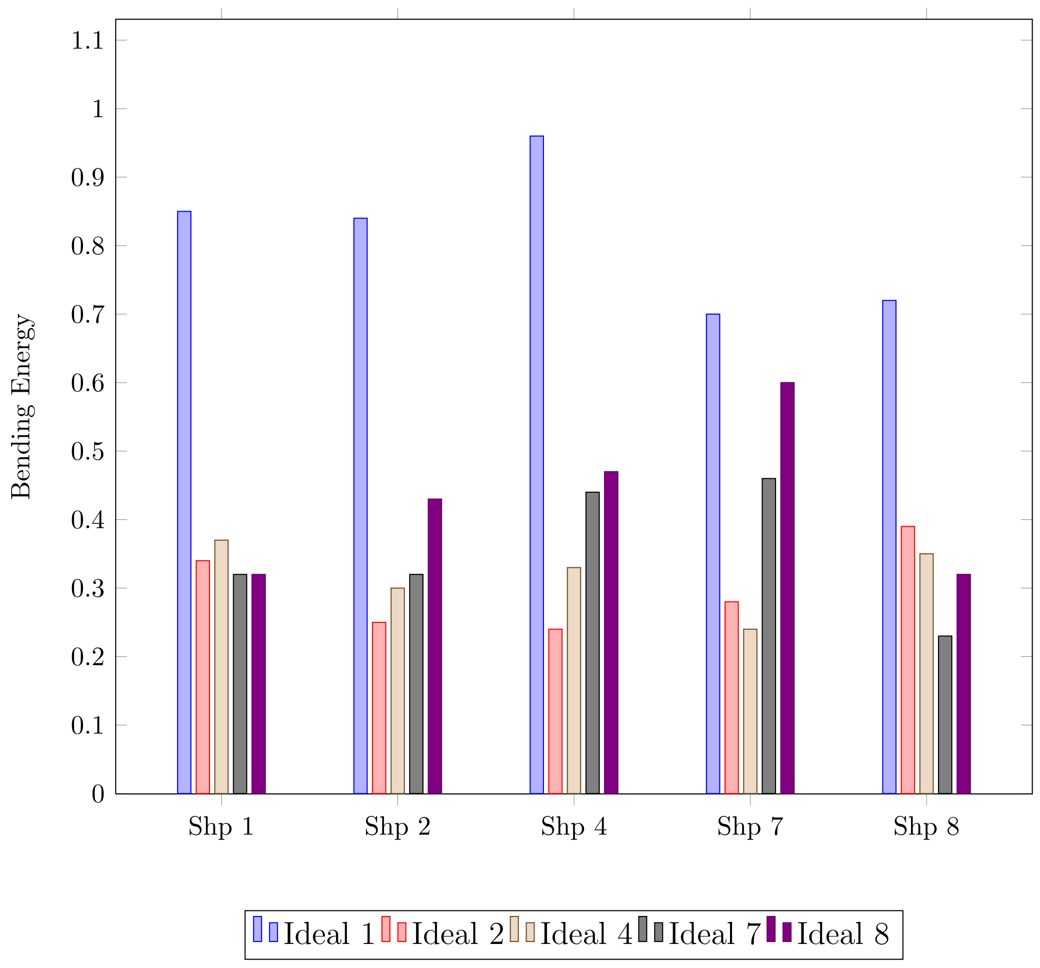


Figure 3.20: Test 2 - Mean bending energy per shape

It confirms some of the results seen in the previous Figure (3.19). Shape 1 has the worst performance of them all. Shapes 2 and 4 have similar performances too, they work well. But Shapes 7 and 8 vary their behaviour. Their values are not close to be the lowest ones when analysing the diagonal.

As was mentioned before, the best way to work with these two parameters should be analysing them together: it could happen that, for example, the error is low (good) but then the bending energy is high (bad). That means that something has gone wrong and the match is not valid. Both error percentage and bending energy should be low in order to obtain a true match.

Test 3

Test 3 was done by two users (1 and 6) on an outdoor environment. Only one scenario was tested. The quality of the pictures is not good enough, therefore the results might not be perfect. Still, they're going to be analysed to see how the experiment has gone.

On Table 3.17 it can be seen the error percentage that each user has. It can be seen that the values are really high: the minimum is 4.50% (User 1 - Ideal Shape 7) and the maximum 7.44% (User 2 - Ideal Shape 1). That indicates that this test wasn't good enough.

Shape 1 is the shape that works the worst, 6.35% of error for User 1 and the already mentioned 7.44% for User 2. Curious is the case of the rejected shapes from previous tests, Shape 3, Shape 5 and Shape 6. It could have been expected for them to have the highest values, but this is not the case. In fact, for User 2, Ideal Shape 5 and 6 have the minimun, 4.58%.

	User 1	User 2
Ideal 1	6.35	7.44
Ideal 2	4.94	5.35
Ideal 3	5.49	5.16
Ideal 4	4.60	4.84
Ideal 5	4.67	4.58
Ideal 6	5,02	4,58
Ideal 7	4,50	4,70
Ideal 8	5,05	5,35

Table 3.17: Test 3 - Medium error (percentage) per user

Table 3.18 shows the results for bending energy in Test 3. Results for

User 1 could be good: except for one Shape (Ideal 1 - 0.37), the values are low, with a maximum 0.25. On the other hand, values for user 2 are very high: Ideal 1 has 1.25, Ideal 4 has 1.01 and so on.

	User 1	User 2
Ideal 1	0,37	1,25
Ideal 2	0,12	0,37
Ideal 3	0,18	0,51
Ideal 4	0,19	1,01
Ideal 5	0,15	0,46
Ideal 6	0,25	0,34
Ideal 7	0,16	0,45
Ideal 8	0,21	0,76

Table 3.18: Test 3 - Bending enery (percentage) per user

From Table (3.19) we can study the validity of the matchings done. Values from the diagonal should be the lowest or be among the lowest of their respective rows and columns.

Except for values from Shape 8, all values from different shapes are very similar. The values for the error percentage are low, very few are higher than 2%. This, in previous tests was a good indicator. In this case, knowing that the pictures have low quality (due to the fact that they were taken in an outdoor environment) doesn't mean the matches are correct. In fact, all values (for all shapes) don't differ too much, so they could not be used for matching.

	Shp 1	Shp 2	Shp 3	Shp 4	Shp 5	Shp 6	Shp 7	Shp 8
Ideal 1	2,26	2,38	2,63	2,53	2,39	2,51	2,55	1,13
Ideal 2	1,96	2,06	1,88	1,82	1,76	1,77	1,73	0,75
Ideal 3	1,71	2,00	1,91	1,83	1,80	2,07	1,96	0,92
Ideal 4	1,75	1,79	1,68	1,66	1,67	1,62	1,67	0,75
Ideal 5	1,70	1,79	1,72	1,72	1,53	1,59	1,58	0,70
Ideal 6	1,79	1,66	1,62	1,74	1,71	1,70	1,79	0,81
Ideal 7	1,82	1,69	1,62	1,57	1,53	1,61	1,62	1,13
Ideal 8	1,84	1,92	1,96	1,90	1,86	1,89	1,71	0,77

Table 3.19: Test 3 - Mean error per shape

Table 3.20 contains the results for bending energy. As can be seen, no Shape is clearly better than any other. But there is one shape that is clearly worse than the others: Shape 1. No clear result can be extracted from this analysis.

	Shp 1	Shp 2	Shp 3	Shp 4	Shp 5	Shp 6	Shp 7	Shp 8
Ideal 1	0,22	0,47	0,23	0,19	0,17	0,22	0,25	0,07
Ideal 2	0,08	0,16	0,18	0,15	0,09	0,07	0,08	0,01
Ideal 3	0,10	0,09	0,08	0,08	0,10	0,12	0,13	0,04
Ideal 4	0,18	0,37	0,23	0,16	0,10	0,09	0,08	0,02
Ideal 5	0,09	0,23	0,21	0,15	0,08	0,05	0,04	0,02
Ideal 6	0,10	0,09	0,10	0,10	0,10	0,09	0,08	0,04
Ideal 7	0,09	0,16	0,16	0,12	0,12	0,13	0,08	0,02
Ideal 8	0,15	0,13	0,17	0,18	0,16	0,10	0,11	0,03

Table 3.20: Test 3 - Mean bending energy per shape

3.6 Results

This section will summarise all the results extracted from the different tests (1, 2 and 3) and from the different methods (Homography and Shape Matching) and presented in section 3.5. Each test will be studied, and comparisons will be made between each method.

Test 1

When working with Homography, Shape 2 offers the best results, with a total of 76.67% of homographies. Shape 7 and Shape 4 perform well to, with 51.11% and 53.33% of shapes recognised.

It would have been expected that Shape 1, creating the biggest glint, would have worked really well. This is not the case, as it is only able to generate 17.78% of homographies.

Shapes 3, 5 and 6 kept Matlab crashing, so they were discarded. The reasons:

- Shape 3. When getting the feature points from the Ideal Shape, 3 or more were found collinear. For this reason, the homography could not be calculated.

- Shape 5 and Shape 6. In the array, they are the ones that use the least amount of LEDs. As a consequence, their glints are very thin, and the minimum of four feature points could not be found. Therefore, no homography was calculated.

It is considered that a Shape performs well if its error is lower than 5%. That's why Shapes 2, 4, 7 and 8 offer good results. Then, if a shape has errors larger than 5%, it indicates they are not good for Shape Context matching. That is the case of Shape 1, with more than 6% of error, joined by Shape 3 and Shape 6, with almost all values above 5%.

When talking about bending energy, it could be stated that a good value, a good match, happens when the value is under 0.20. In exceptional situations, that limit could be raised up to 0.25, but that's a threshold that should not be used often. Having said this, Shape 2 and Shape 8 both have 0.09 in the diagonal (see Table 3.12). Shape 7 also has good value (0.15) and Shape 4 is on the limit with (0.21). Shape 1, normally with bad results, this time is under the limit (0.20). But Shape 3 (0.36) and Shape 6 (0.37) have performed very badly.

It has been already been mentioned that when working with Shape Context, the best way to have a robust match is to combine the results from the error percentage and bending energy. Following this reasoning, the shape that works better is Shape 8. Shape 2 and Shape 7 also have a good performance. The shape that shouldn't be used based on this reasoning is Shape 1.

Taking into consideration both methods (Homology and Shape Context), it is clear that Shapes 8, 2 and 7, even 4, are the ones that have better results (in that order). Another thing that's clear is that Shapes 1, 3, 5 and 6 should not be used for Homology and/or Shape Context.

Test 2

When using homographies, in this test, the one that works better is Shape 7, with 19.9% of homographies. Shape 8 and Shape 2 have good results too, 13.99% and 13.78% respectively. Shape 1 offers again the worst results, with as little as 5.78% of homographies obtained.

If the analysis is made looking at skin colour, there are no big differences in the performance of the shapes: 2, 4, 7 and 8 have good results, and 1, 3, 5 and 6 not so good. The same thing happens if the focus is put on eye colour.

Looking at the error percentage on Shape Context matching, it is confirmed that Shapes 4 and 7 work well, no matter what skin colour is being analysed. Shapes 5 and 6 keep having the biggest errors, as does 1. Shape 3 has a mixed behaviour: sometimes is over 5% and sometimes it is less than 5%, not a good match. If the discussion is about eye colour, results are in the same line as before.

Now, dealing with bending energy: Shape 1 works the worst. Four out of 6 values are higher than 0.5, with one of them as high as 2.89. In general, bending energy values are lower than 0.5, even lower than the desired 0.2. Paying attention to skin colour and eye colour, it can be seen again that Shapes 2, 4 and 7 have the best performances. In most of the cases, their values are around 0.10-0.15 (less than 0.2). In this occasion, Shape 8 doesn't perform well. Shape 1 is, once more, the worst.

Studying the performance of the shapes independently, is is obvious that Shape 7 offers the best results, with lower error percentages, and differences between them of less than 1%. On the other hand, Shape 1 has values over 5% and even 6%. On this occasion, the performance of Shapes 7 and 8 is not as good as Shape 2 and 4. But, Shape 1 is the worst.

If we compared all methods together, it is clear that Shape 7 is the one that should be used in the first place, since its performance is the best. Other options would be 2, 4 and 8. Shapes 1, 3, 5 and 6 are discarded this time too due to their poor behaviour in the matching process.

Test 3

It has already been said that the quality of the pictures is not good enough, with this being the reason for such bad results. Nevertheless, the data has been analysed too.

Since the pictures were low quality, not many glints could be recognised. So, no more than 6% of homographies could be obtained from them. In fact, one of the users taking the test had only 1.1% of homographies, in just one of the shapes (the rest were 0%).

The error in this test is high, it is more than 5% in all cases. The maximum reached is 7.44% error. It is curious that Shapes 3, 5 and 6, normally with poor performances, don't show values which are too high. Well, at least they are on the same level as the rest of the shapes.

Paying attention only to bending energy, for User 1 they are almost all lower than 0.25, which is a good indicator. User 6, on the other hand, has no good values at all.

If both error percentage and bending energy are combined, to get a more robust matching, no conclusion can be made with these values. The ones from bending energy are good, but not the ones from the error percentage.

Values for mean error per shape and mean bending energy per shape are small. At first, this could mean something good. But since the photos were bad quality, these results would require a deeper analysis.

The poor quality of the pictures is keeping the methods from obtaining good enough results.

Chapter 4

Conclusions

In this work, a study has been carried out in order to find robust methods for mobile eye tracking. A new approach was performed: object recognition applied to mobile eye tracking. Instead of using individual LEDs (with a maximum of five), an 8x8 IR LEDs array was built, and turned on different shapes in order to generate shaped glints on the eye. Those shaped-glints were used as input to two methods: Homography and Shape Context matching. Three tests were done in order to evaluate both the performance of the shapes and the two methods. Up to six users sat through a maximum of six scenarios, from which five pictures were taken, totalling 800 pictures processed by Homography and Shape Context. MATLAB has been used for processing the data, modifying available code to meet the requirements of this project and also writing new code.

A number of conclusions have been reached during the course of the study. There are no significant differences in the results for users with different skin colour, as users with pale skin showed similar results to those with darker skin. Also, there are no significant differences for users with different eye colours. Therefore, the methods studied are invariant to skin and eye colour. Concerning the shapes, it has been found that thin shapes are not appropriate for this way of working, as they could not be processed (MATLAB crashed when processing these shapes). Also, it has been found that big shapes such a square, or a half a square (right-angled triangle) didn't perform so well. On the other hand diamond, isosceles triangle and arrow shapes had the best performance, having in common a pointed edge and a wide body. So shapes with pointed edges and wide bodies are expected to perform well, but shapes with thin or very large bodies won't be of any use. Although experiments were carried out in an outdoor environment, no conclusion can be reached, due to the low quality of the pictures taken. If higher quality pictures were at hand, they should have given results very similar to the ones from indoor tests.

In addition to these conclusions, it was found that there should be more than one indicator in order to say if a match is correct or not. Homography method only provided one indicator; Shape Context provided two. Combining these two indicators (error and bending energy) yielded better results, as "false positives" were discarded and at the same time correct matches were confirmed twice.

Further experiments and investigations could be done in order to find more indicators that could work with Homography, in order to make it more robust. Also, once it's been narrowed which shapes have good results, new ones could be designed in order to reach the one that performs the best.

Bibliography

[1] Arduino. http://arduino.cc/playground/main/max72xxhardware.

[2] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *Transactions on Pattern Analysis and Machine Intelligence*, 24(24), 2002.

[3] Andrew T. Duchowski. *Eye Tracking Methodology: Theory and Practice.* Springer.

[4] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.

[5] D.W. Hansen and Q. Ji. In the eye of the beholder: A survey of models for eyes and gaze. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):478–500, 2010.

[6] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge, 2 edition, 2003.

[7] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing", howpublished = "Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[8] Javier San Agustn L. *Off-the-Shelf Gaze Interaction.* PhD thesis, IT-University of Copenhagen, 2009.

PORTADA

Index

- Introduction
- Theoretical Principles
- Experiment
- Analysis & Results
- Conclusions

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

IT University of Copenhagen

INTRODUCTION

Introduction

- Eye Tracking: the process of measuring the point of gaze or the motion of the eye.
- Application:
  - Diagnostic
  - Interactive
- Types of eye trackers:
  - Non - mobile
  - Mobile
- Goal: Find a solution robust enough for mobile eye trackers.

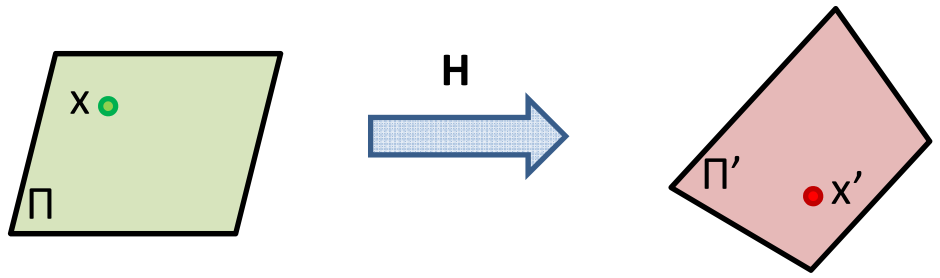
Introduction

- Novelty approach:
  - Object recognition
  - Shaped-glints
- Methods:
  - Homography
    - No calibration
    - Higher accuracies
  - Shape Context matching
    - Used widely
- Compare their performances

THEORETICAL PRINCIPLES

Theoretical Principles - Homography

- Homography: projective transformation that maps points from one plane to another plane.



- Simple equation:

$$x' = Hx$$

- Examined more closely:

$$H = \begin{bmatrix} A & t \\ V^T & v \end{bmatrix}$$

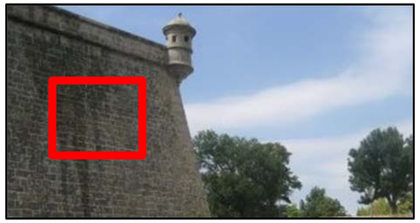
A → Rotations and non-isotropic scaling

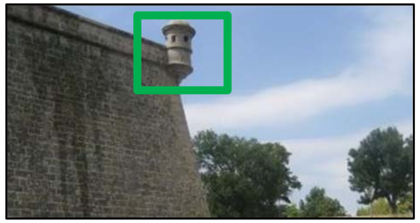
t → Translations

V^T → Non-linear effects

Theoretical Principles - Harris Corner Detector

- Detects feature points: corners
- Strong invariance to:
  - Rotation
  - Scale
  - Illumination variation
  - Image noise
- Window:


No Variation

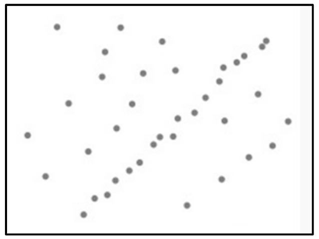

Variation

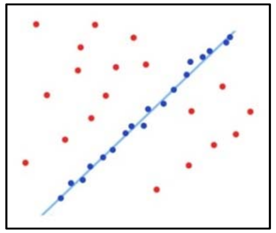
- Mathematically:

$$\boxed{E(u,v)} = \sum_{x,y} \boxed{w(x,y)} \left[\boxed{I(x+u,y+v)} - \boxed{I(x,y)} \right]^2$$

Theoretical Principles - RANSAC

- Extracts from a set of outliers the inliers fitting a model:


Set of Outliers


Fitted Line

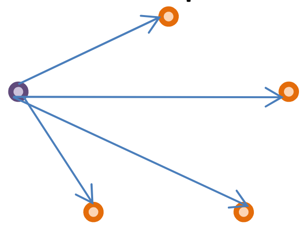
- Minimizes the mismatches.

- Gives more robustness.

Theoretical Principles - Shape Context

- Measures the similarity between two shapes
- Simple process:
  - Find correspondences between two sets of points
  - Estimate an aligning transform
- Works with edges. Sample points are extracted from them.
- Process:
  - Takes a point p_i on one shape.
  - Takes the coordinates of the remaining (n-1) points.
  - Calculates a histogram h_i of the relative coordinates. That histogram is the shape context for point p_i.

  Repeated for all the points on both shapes.



Theoretical Principles - Shape Context

- Two indicators are extracted:
  - Error: Mean squared error between synthetic warped image and estimated warped image.
  - Bending Energy: Amount of transformation necessary to align the shapes.

- mj

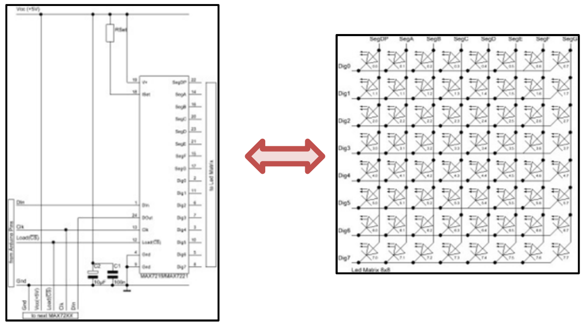
EXPERIMENT

Experiment

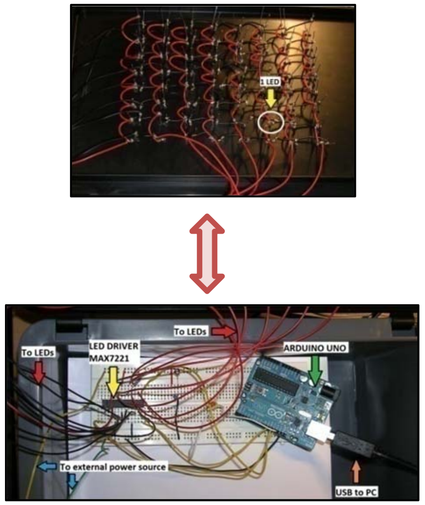
- Interoperability: 3 different programs / devices working together.

$$\begin{tikzcd}\boxed{\text{Matlab}} \arrow[r, leftrightarrow] & \boxed{\text{Arduino}} \arrow[r, leftrightarrow] & \boxed{\text{MAX7221}}\end{tikzcd}$$

- Connections



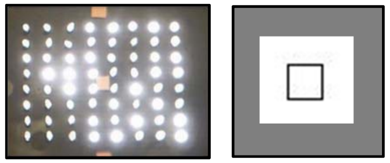
Schematics

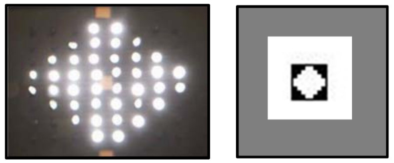


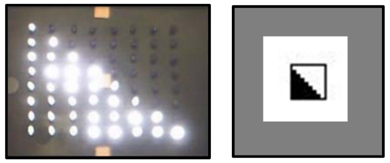
Real connections

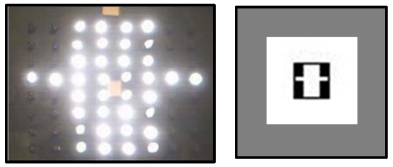
Experiment – Shapes

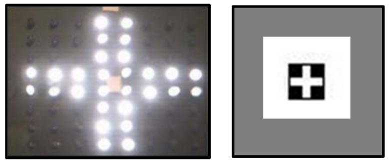
- Shapes: One of the foundations of this project.
- How to choose them:
  - Recognizable shapes.
  - Different types of shapes.
  - Enough quantity.
- Total = 8 Shapes: 2 edge shapes + 6 full shapes

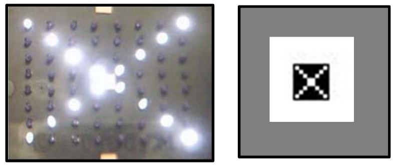

Shape 1

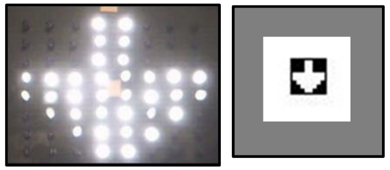

Shape 2

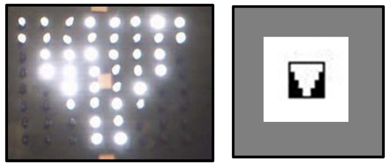

Shape 3


Shape 4


Shape 5


Shape 6


Shape 7


Shape 8

Experiment – Scenarios & Test definitions

- 2 main scenarios were analyzed: Outdoors and Indoors.
- 3 different tests were carried out.
- 5 points to look at.
- 8 Shapes.

	# Scenarios	# Users	# Points/shape	# Pictures
INDOORS				
Test 1	6	1	5	240
Test 2	2	6	5	480
OUTDOORS				
Test 3	1	2	5	80
TOTAL				
				800

Experiment – Users

- Variety of skin and eye color, as well as different eye morphology.
- Aim: analysing the performance of the methods from different users.

User 1: Pale Skin
Dark brown eyes

User 2: Pale Skin
Brown eyes

User 1: Pale Skin
Light brown eyes

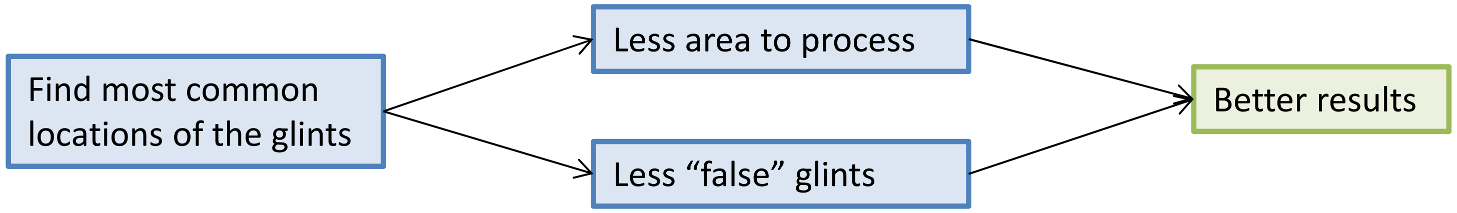
User 1: Dark Skin
Dark brown eyes

User 1: Pale Skin
Green eyes

User 1: Pale Skin
Light brown eyes

Experiment – Preparatory work

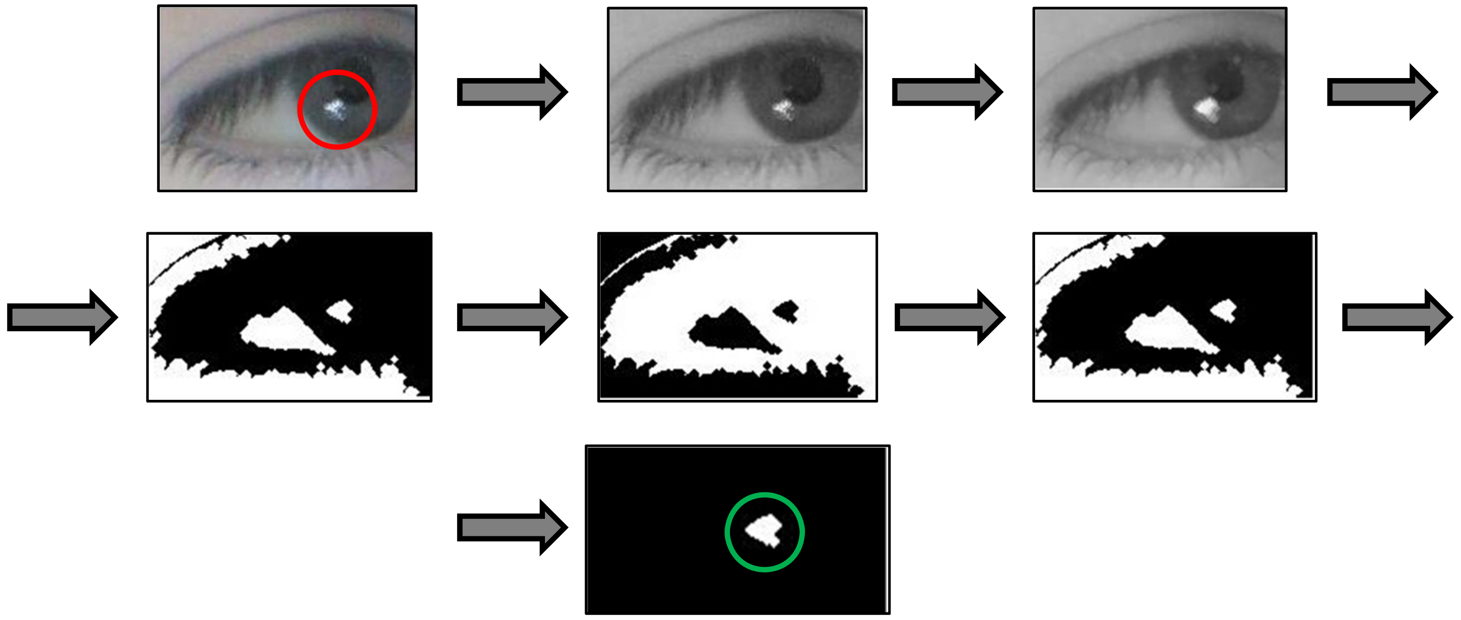
- Early tests were done in order to:



- Ideal databases were created for both methods using the ideal shapes.
  - Homography: Feature points (corners) were detected in the ideal shapes. Their coordinates, saved in a database.
  - Shape Context: The edges of the ideal shapes were detected; their coordinates, saved in a database.

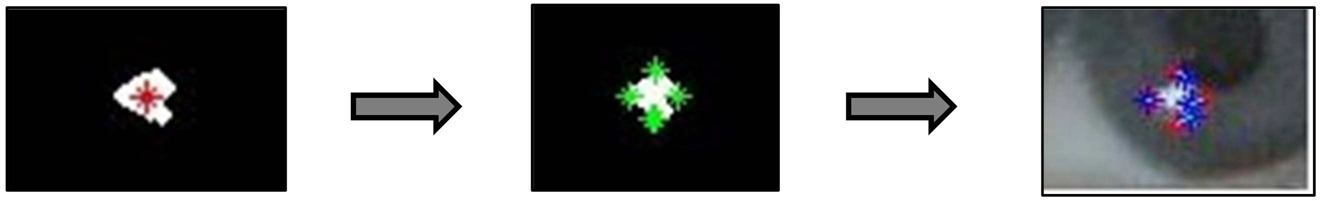
Experiment – Tests

- Preparatory work: Ideal databases were created for both methods using the ideal shapes.
  - Homography: Feature points (corners) were detected in the ideal shapes. Their coordinates, saved in a database.
  - Shape Context: The edges of the ideal shapes were detected; their coordinates, saved in a database.
- The image is pre-processed:

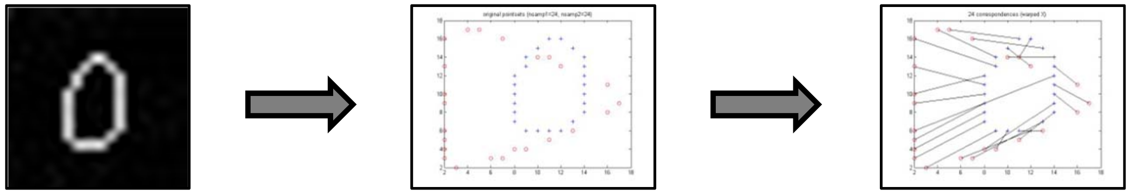


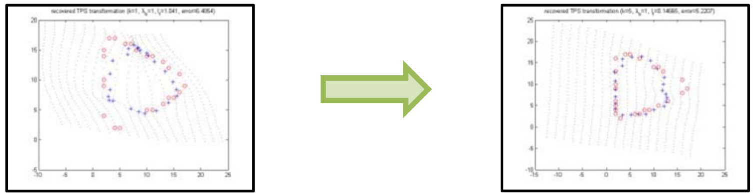
Experiment – Tests

- Homography:



  - Homography.
  - Percentage.
- Shape Context:



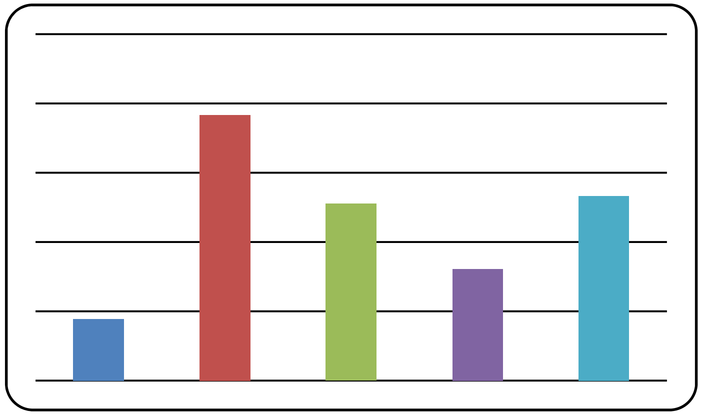


- Error
- Bending Energy

ANALYSIS & RESULTS

Analysis & Results – Test 1 - Homography

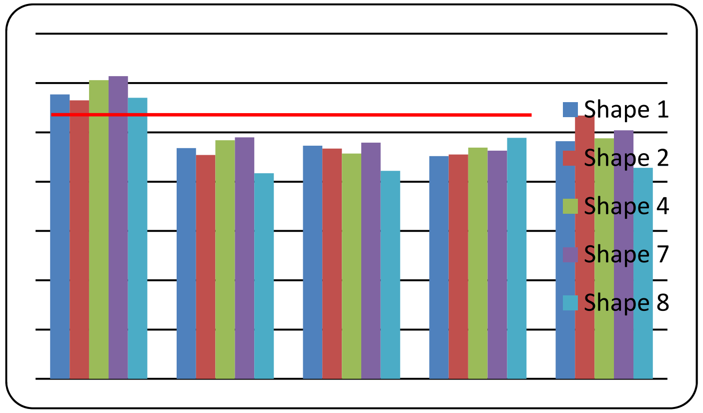
- Homography


Homography (%)

- Shape Performance
  - ✓ Shape 2 (76.67%), Shape 4 (51.11%), Shape 8 (53.33%).
  - ✘ Shape 1 (17.78%), Shape 7 (32.22%).

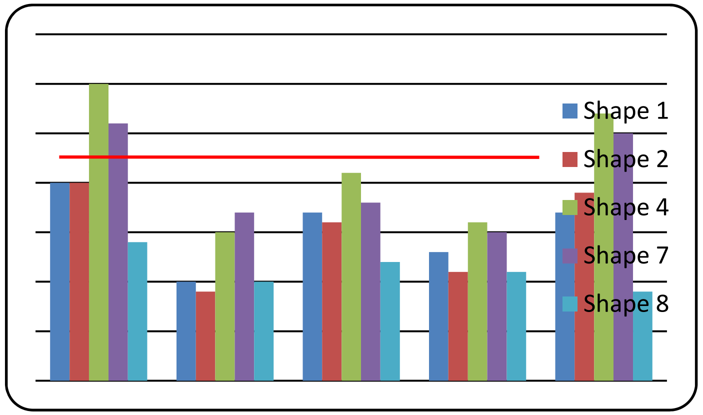
Analysis & Results – Test 1 – Shape Context

- Shape Context



Error (%)

- Error (%)
  - ✓ Shape 2, Shape 4, Shape 7, Shape 8.
  - ✘ Shape 1.

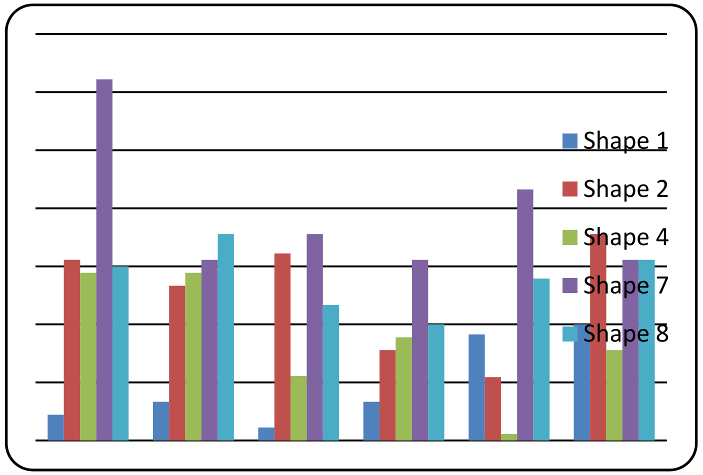


Bending Energy

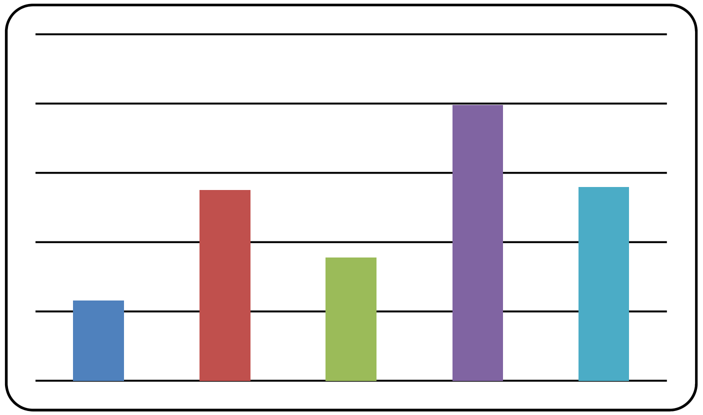
- Bending Energy
  - ✓ Shape 2, Shape 8
  - → Shape 4, Shape 7
  - ✘ Shape 1

Analysis & Results – Test 2 – Homography

- Homography



Performance of the shapes (%)



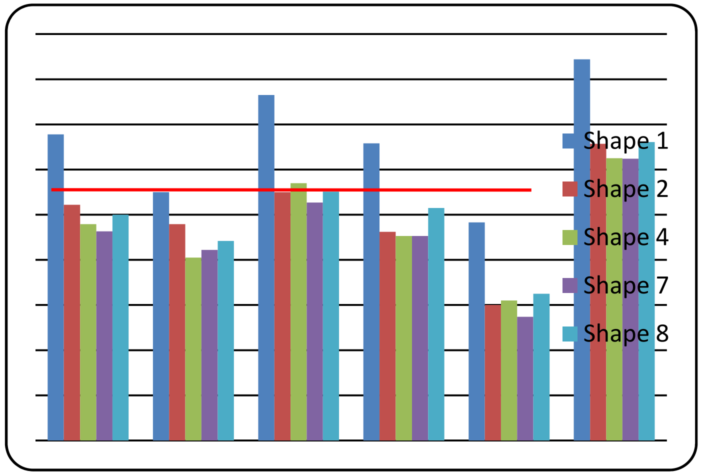
Homographies – mean value(%)

- Users Performance:
  - Skin color: No significant differences.
  - Eye color: No significant differences.

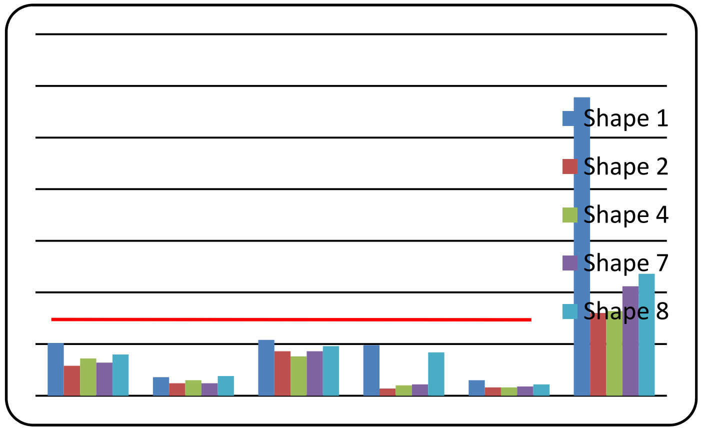
- Shape Performance
  - ✓ Shape 7 (19.9%).
  - → Shape 2 (13.78%), Shape 8 (13.99%).
  - ✗ Shape 1 (5.78%), Shape 4 (8.89%).

Analysis & Results – Test 2 – Shape Context

- Shape Context



Error(%)

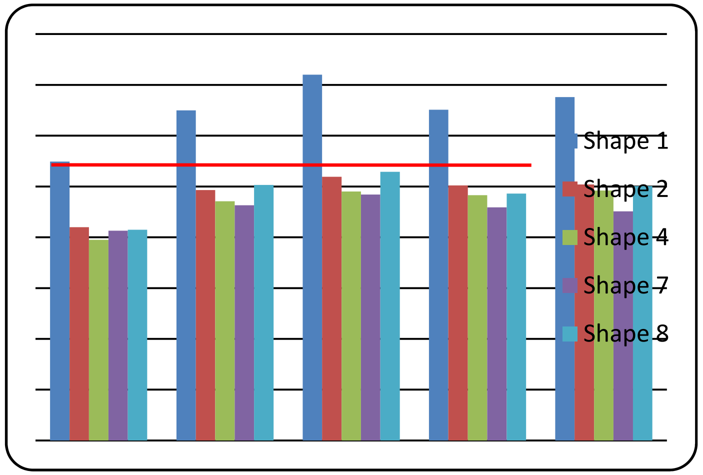


Bending Energy– mean value

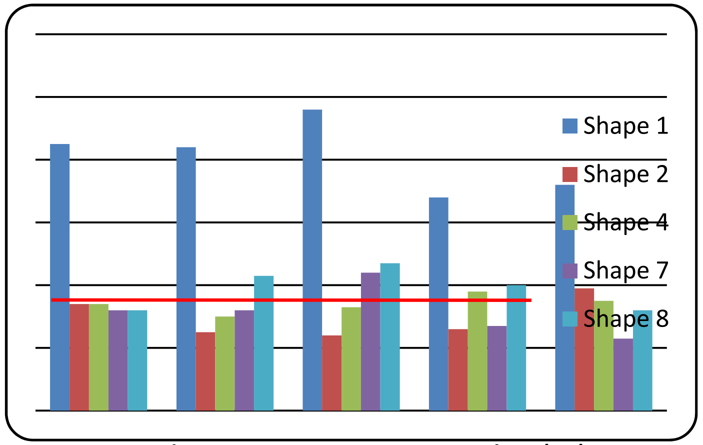
- Users Performance (based on Error):
  - Skin color: No significant differences.
  - Eye color: No significant differences.
- Users Performance (based on Bending Energy):
  - Skin color: No significant differences.
  - Eye color: No significant differences.

Analysis & Results – Test 2 – Shape Context

- Shape Context



Performance of the shapes (%)

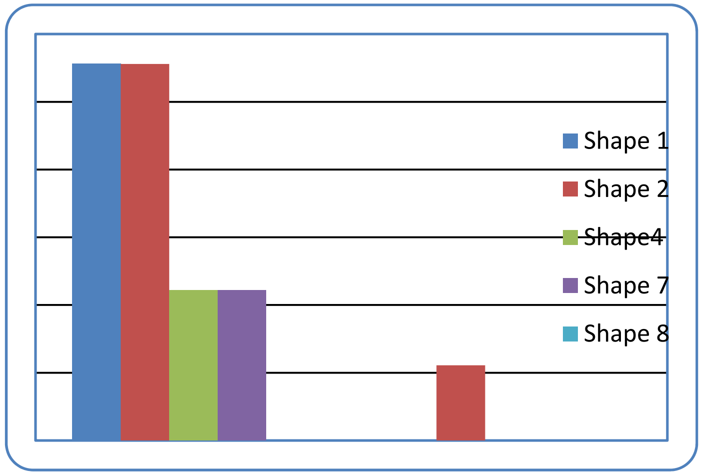


Bending Energy– mean value(%)

- Shape Performance (based on Error):
  - Skin color: No significant differences.
  - Eye color: No significant differences.
- Shape Performance (based on Bending Energy):
  - Skin color: No significant differences.
  - Eye color: No significant differences.

Analysis & Results – Test 3

- Homography

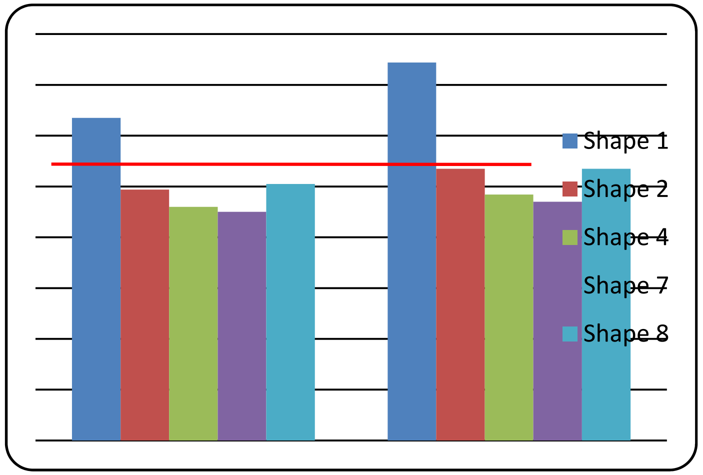


Performance of the shapes (%)

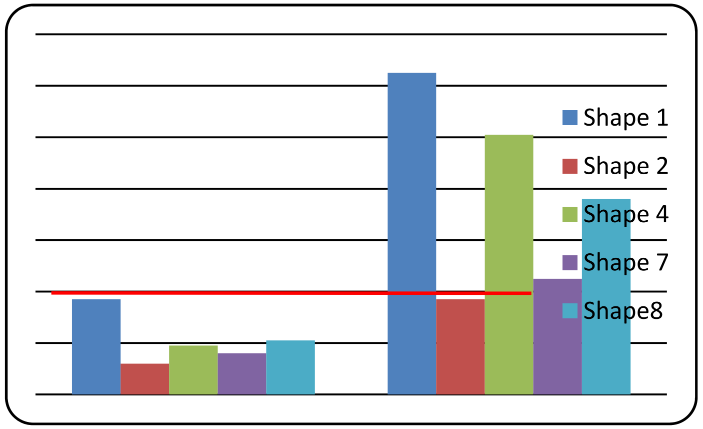
- Shape performance:
- highest percentage is 5.56% → No conclusion.

Analysis & Results – Test 3 – Shape Context

- Shape Context



Error(%)



Bending Energy– mean value

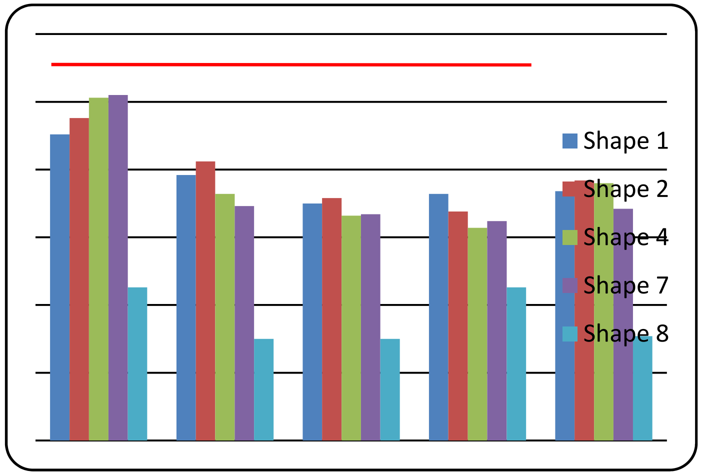
- Users Performance (based on Error):BLABLA
  - Skin color: No significant differences.
  - Eye color: No significant differences.

BLABLA

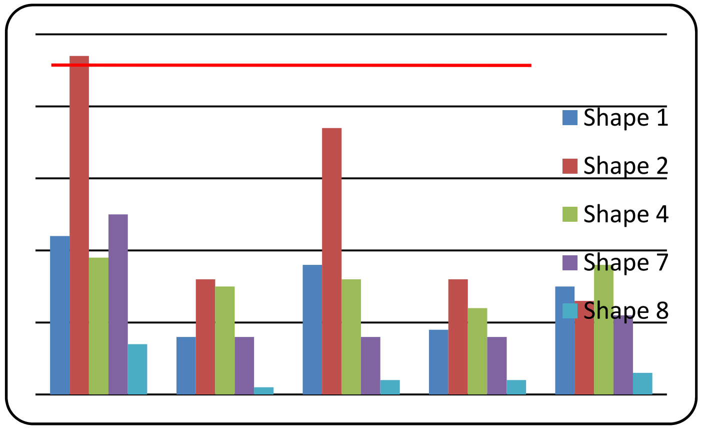
- Users Performance (based on Bending Energy):
  - Skin color: No significant differences.
  - Eye color: No significant differences.

Analysis & Results – Test 3 – Shape Context

- Shape Context



Performance of the shapes (%)



Bending Energy– mean value(%)

- Shape Performance (based on Error):
  - Skin color: No significant differences.
  - Eye color: No significant differences.
- Shape Performance (based on Bending Energy):
  - Skin color: No significant differences.
  - Eye color: No significant differences.

CONCLUSIONS

Conclusions

- No significant differences for users with different skin colour.
- No significant differences for users with different eye colour.

➡ Invariant to skin and eye colour.

- Thin shapes are not appropiate.
- Big shapes (like square or similar) don’t have good performances.
- Diamond, isosceles triangle, arrow shapes offer good results.

➡ Shapes with pointed edges and wide bodies give the best performance

- No conclusions from outdoor test.

THANK YOU
VERY MUCH

QUESTIONS?