# Coin Recognition Project

***An image pattern recognition project for mobile devices***

Telecommunications Systems Institute
Supervisor: Papaeystathiou Yiannis
Author: Kalodimas Panagiotis

# CONTENTS

# FIGURES

# INTRODUCTION

This report is related to an internal project, called “Coin Recognition Project” behalf of Telecommunication Systems Institute. The intention of the project was to examine the possibility of applying useful image pattern recognition applications in embedded devices with the capabilities of the modern mobile phone devices, known as smart phones. The application was developed in that scope tries to recognize the identity of the coins that a user can capture by its smart phone camera. The main constraints need to be overcome for this purpose is low frequency processors and the lack of memory available in this kind of devices. The development of the application was the main concept of this research. This is the basis for experimental processes and the improvement proposals.

# ABSTRACT

The basic keys for the implementation of the Coin recognition project application was the pattern recognition algorithm to be used and the mobile device on which the development had to be on. The "2D Normalized Cross Correlation" was the pattern recognition algorithm that was chosen and the Apple iphone 4G was the mobile device. The first chapter is dedicated to the analysis of the "2D Normalized Cross correlation" algorithm and its custom implementation used in this project. In the second one, there will be a short statement about the Apple iphone 4G smart phone and the iOS operating system on which the project's application was developed. After the reference on these basic keys of the project, in the third chapter there is an analytical presentation of the iOS application that was implemented and can be found in Apple AppStore to be installed on any iOS device.

# CHAPTER 1

# 2D NORMALIZED CROSS CORRELATION ALGORITHM

This section describes the "2D Normalized Cross Correlation" object recognition algorithm and its custom implementation used in "Coin Recognition Project". The "2D normalized Cross Correlation" algorithm is an expansion of the "Normalized Cross Correlation" one used in digital signal processing modified for images. The reason this algorithm was chosen is the advantage it gives as long as its complexity and its performance. It is a simple and fast algorithm with good enough recognizing results.

## 1.1. Correlation & 2D Correlation

Correlation is a technique used on creating an associating representation between two different signals. It shows the way these signals affect its other when they conflict. 2D (2 Dimensional) correlation is a patch used for measuring image signal correlation. The correlation is measured by multiplying the two signals each other changing the factor of the collision time. The collision may occur any time during a signal emission. Correlation method represents the total effect of the collision in any possible time. It is like sliding the two signals each other and measuring its association in every slide size.

$$R_{xy}(n) = \frac{1}{N}\sum_{i=1}^{N-n} x_i \cdot y_{i+n}$$

$R_{xy}(n)$ is the correlation of the two signals in the factor of time n.

The $\frac{1}{N}$ factor causes normalization

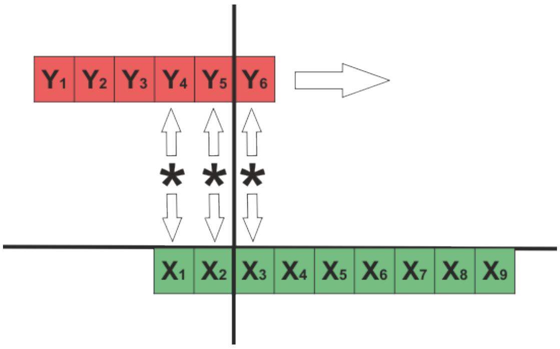



*Figure 1: Correlation* $R_{xy}(3)$

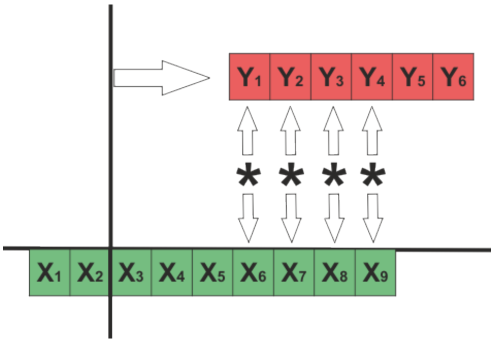



*Figure 2: Correlation* $R_{xy}(11)$

For example, in Figure 1, for time n=3, $R_{xy}(3) = \frac{1}{6}(X_1 * Y_4 + X_2 * Y_5 + X_3 * Y_6)$

The Two Dimensional (2D) correlation is a similar approach with the difference of having two dimensional signals (images). As the one dimension correlation, in the 2D one the one image slides over the other correlating each of their pixels. The Figure 3 makes it simpler to understand.

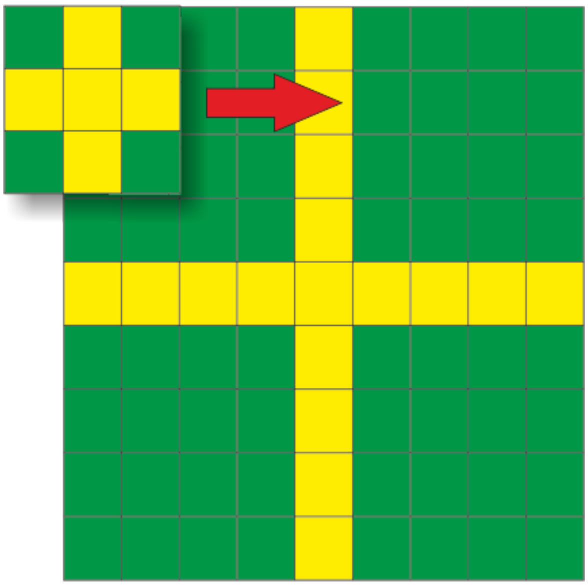


*Figure 3: 2D Correlation*

## 1.2. Normalized Cross Correlation Recognition Algorithm

The “Normalized cross correlation” algorithm is based on the attributes of the correlation of two similar signals. This time each signal value is subtracting the mean value and the size of the correlation is divided by the square root of the two signals deviance. The result is the normalized cross correlation factor $R_{xy} \subset \{-1,1\}$. This factor shows the similarity of the two signals. A value sized of 1 shows 100% similarity. The “normalized cross correlation” algorithm mathematic type is as follows:

$$R_{xy}(t) = \frac{\sum_{i=0}^{N+M} (x_i - \overline{x}) \cdot (y_{i+t} - \overline{y})}{\sigma_x \sigma_y},$$

N is the size of signal x and M the size of y

Given two images where the first one is the main image (sized AxB) and the second one (sized NxM) is the sample that has to be recognized inside the main, mathematically, the 2D correlation operation can be described as follows:

$$R_{xy}(n) = \frac{\sum_{i=0}^{N}\sum_{j=0}^{M}\left(x_{i+a,j+b} - \overline{x}\right)\cdot\left(y_{i,j} - \overline{y}\right)}{\sqrt{\sum_{i=0}^{N}\sum_{j=0}^{M}\left(x_{i+a,j+b} - \overline{x}\right)^2 \cdot \sum_{i=0}^{N}\sum_{j=0}^{M}\left(y_{i,j} - \overline{y}\right)^2}} \qquad a = n / A, b = n \bmod A$$

The function variable *n* replaces the time factor of the signal processing correlation algorithm. Instead of time, “2D normalized cross correlation” algorithm uses the spatial factor of index in the main image frame. For example, when *n* is equal to 0 the upper left corner of the main and the sample image concur. Is the place where the “sliding” process begins. The variables *x* and *y* represents the two images pixels values. The factor *a* and *b* are used to calculate the row and the column that the *n* variable represents in the main’s image pixel array.

As an example, trying to recognize the sample image of the Figure 4 inside a main image in the same figure, the results of Figure 6 comes in using the “2D normalized cross correlation” in Matlab. The lighter area in main image in Figure 4 shows the area matching with the sample image.


*Figure 4: Matlab Experiment – Main & Sample Image*


*Figure 5: Matlab Experiment – Matching Results*

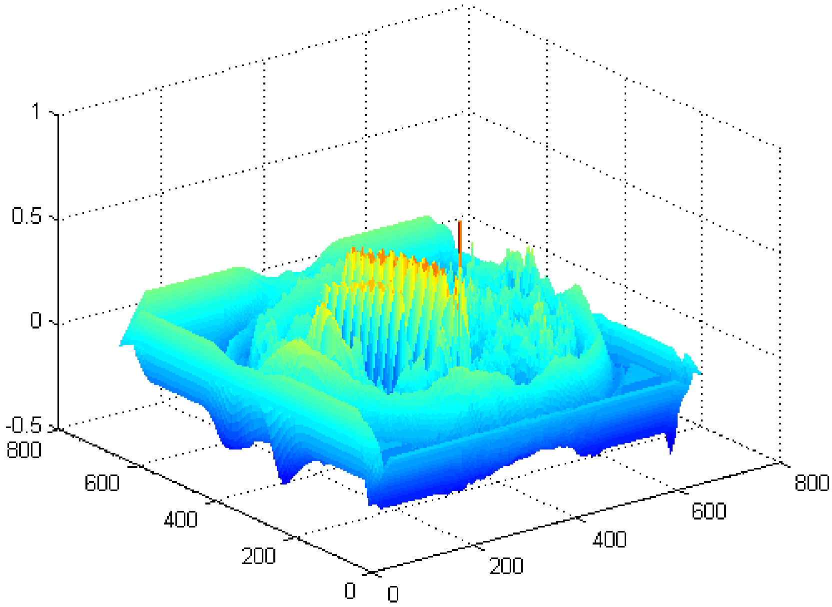



*Figure 6: Matlab Experiment - Correlation Plot*

In the “2D normalized cross correlation” algorithm, the subtraction by the mean value of every pixel creates a brightness normalization that is very important on image processing. The brightness affects the values of the image signal although it does not offer more useful information. The brightness differences between two similar images is one more constraint factor has to be solved. The normalization helps on solving this problem.

Testing the “2D normalized cross correlation” algorithm using the same sample image but different brightness main images gave the results of the following figures. The Figure 7 show the normalized cross correlation results of the image in Figure 8 where the matched sample is pointed lighter. Figure 9 shows the normalized cross correlation results of the lighter image in Figure 10.

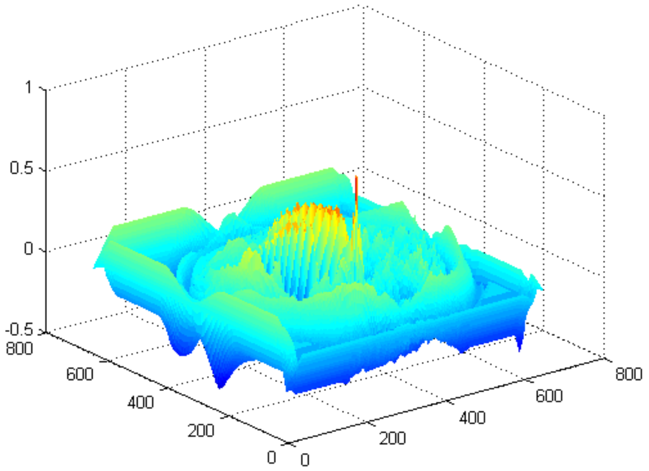



*Figure 7: Matlab - Correlation Plot With Darker Image*


*Figure 8: Matlab - Darker Image*

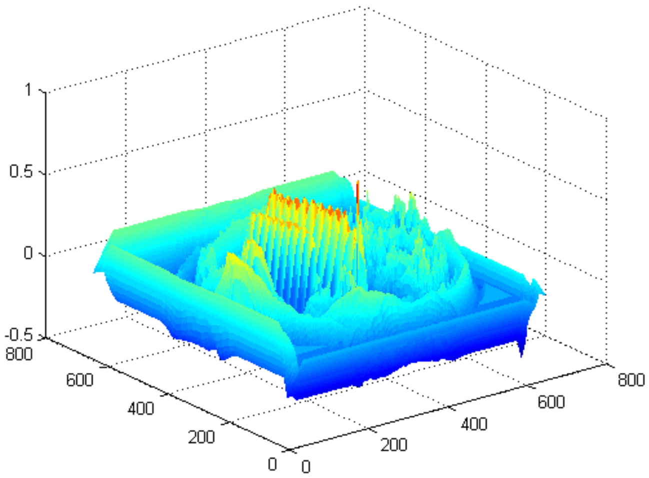



*Figure 9: Matlab - Correlation Plot With Lighter Image*


*Figure 10: Matlab – Lighter Image*

## 1.3. Coins Recognition Project and Normalized Cross Correlation

The «Normalized cross correlation» algorithm is a general usage method for correlation and pattern matching. In the coins recognition project a more specific implementation was design for customizing the process to the certain purpose. The intention of those customizations was the speeding up of the process in the ration of efficiency. In the following subchapters the parameters of the customized implementation are described before referring to the recognition process.

### 1.3.1. Template Sampling

The recognition samples for each coin used for the correlation process are grabbed by a template image for each coin. The template image is divided in square tiles. The template and the samples tiles size are configured by the user in application's «Settings Menu» (Chapter 2.2.1).

*Figure 11: Template Sampling*

### 1.3.2. Sampling Depth

The number of tile-samples grabbed from a coin's template images is decided by the user in the «Settings» menu by the parameter «Depth» (Chapter 2.2.2) which defines the tile distance from the center tile. A template sampled by the depth of 1 produce a tile in the center of the image which corresponds with the center of the coin figuring. The main tile of any template is always the center tile. In Figure 12 the tiles of depth *N* are all the tiles with the number equal and minimum to *N.* In the recognition process, as it will be described in Chapter 1.3.6, it is not certain that all the tiles will be used. The greatest the depth parameter is, more accurate the recognition process will be in negative ration in the data having to be saved in the application database.


*Figure 12: Template Tiles by Depth*

### 1.3.3. Rotation Angle

One more parameter affecting the recognition project is the rotation angle that maybe applied to the coin image when captured by the user. It is supposed that as long as the image capture procedure is controlled by a human the rotation mismatch can be probably reduced to ±15$^{o}$ degrees or less. This parameter can be defined by the «Rotation Angle» option (Chapter 2.2.2). The greater this parameter is the slower the recognition process will be.

*Figure 13: Rotation Angle (30$^{o}$ - Green Line) and Rotation Step (10$^{o}$ - Red Lines)*

### 1.3.4. Rotation Angle Step

Relative to the «Rotation Angle» parameter is the «Angle Step» parameter which defines the rotation angle step in the recognition process. This means that having the «Angle Step» parameter defines as «5$^{o}$» and the «Rotation Angle» parameter as «15$^{o}$», the recognition process will correlate the coin image rotated by 0$^{o}$, ±5$^{o}$, ±10$^{o}$ and ±15$^{o}$ degrees with the template images. The least this parameter is, the better the efficiency will be causing more correlation processes that increase the whole procedure time needed.

### 1.3.5. Padding

As an image is captured by the user, it ensures that between the image center and the coin center a mismatch will occur. This situation can be solved by the «Padding» parameter which defines the maximum possible mismatch. For a «Padding» value of «20» every sample is going to be correlated with a tile lengthen by 20 pixels to all directions.


*Figure 14: Padding – Sample tile (Green) and Correlation tile (Red)*

### 1.3.6. Accuracy

The «Accuracy» parameter is a custom parameter used in the recognition process in order to speed it up. Assuming a large database of coins templates to be correlated with the one to be recognized, the rejection of the most uncorrelated templates is intended. After correlating a coin's tile with all templates corresponding tiles a first outlook about which templates are more alike to the coin template conclude. This information can cause the inclusion of those templates in the next sample correlation process rejecting the rest ones. This technique can increase the accuracy of the recognition process and reduce the time needed. The «Accuracy» parameter defines the similarity value need to be achieved by a template to qualify to the next tile correlation.

### 1.3.7. Recognition Process

After explaining the parameters of the recognition process it is feasible to explain the coin's recognition process using the «Normalized Cross Correlation Algorithm».

First of all, a preparative process occurs by the time the user captures the coin's image using the device's camera or picking it from the photo library. The image quality is much under the responsibility of the user. The user has the ability to edit the image before the recognition process begins. When the user provides the image for recognition the application is automatically resizing the image to the template images size and converting it to a black & white image using 8 bit information for each pixel.

1. The application uses a local correlation strategy. It does not correlate any sample with the whole coin's image but instead it crops the coin's image in tiles as it does with the template images. The tiles size is defined by the template images tiles and the padding parameter. The first tile cropped is always the main tile (Tile in the center of the image).

2. After all templates main tile is correlated with the coin's correspond tile, the coin's image is rotated by an «Angle Step» parameter angle and the previous process (stage 1) is repeated. This process is repeated for all the angle steps until the «Rotation Angle» is reached.

3. By this point all templates main tile is used for recognizing the coin overcoming any padding and rotation mismatch. On this stage the application uses the greatest correlation values occurred in each template (independent of rotation and padding) to reject the minimal values and keep the highest. The templates that will be used in the next stage are those whose correlation result is higher than the «Accuracy» parameter.

4. The previous stages (1-3) will be repeated until the correlation result of only one template sample is larger than the «Accuracy» parameter of all the others or all the templates samples have been used. In the second case, the template with the large correlation mean value is defines as the matched one. The samples usage hierarchy is illustrated in Figure 15.

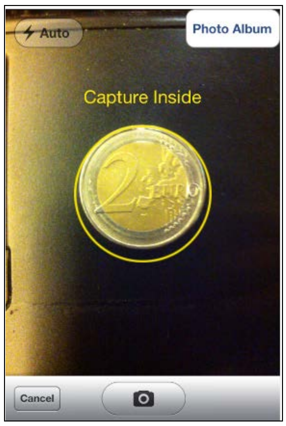

*Figure 15: Sample Usage Hierarchy*

As it is explained in the recognition process, the application tries to reduce the correlations applied by cropping the image to smaller pieces and rejecting templates during the process. The Figure 16 illustrates the process describes so far.

## Preparative Stage

**Pick Image From Photo Library**


**Capture Image with Camera**



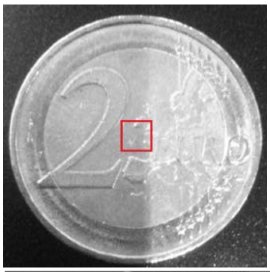
**Manual or Auto Crop Image to the frame of the coin**


**Resize the image according to the image settings «template Size» (default 140x140 pixels)**

**Convert image  to Black and White color format**

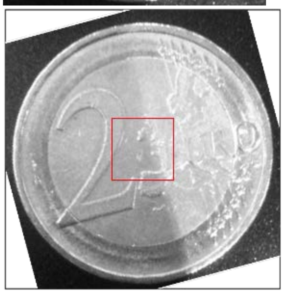
## Correlate Stage

**From Tile 1 to N (according to  «Depth» option)**

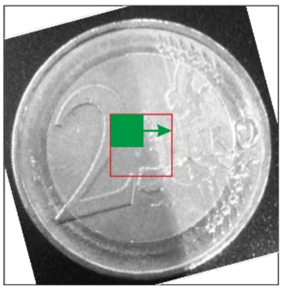


**From -«Angle» to «Angle»**
**with step «Angle Step»**

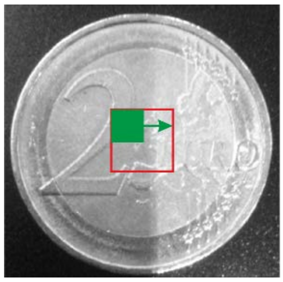
1. **rotate image to Angle degrees**
2. **crop image to tile N with padding**



3. **correlate cropped image tile with coins corresponding tile**



**Repeat stages 1 to 3**



4. **Pick Over Stage**

1. Find coin with maximum correlation
2. Remove coins with correlation less than maxCorrelation*«Accurancy» from list
3. If only one coin left in the list go to Recognition Stage else repeat the Correlate stage for the next tile

## Recognition Stage

- **If only one coin left after «Pick Over» Stage then that's the coin recognized**
- **If more than one coins are still in queue then pick the one with the maximum mean correlation value**


*Figure 16: Image Recognition Process Stages*

## 1.4. Coins Recognition Implementation

The implementation of the project is set on the iPhone 4G mobile device, developed by "Apple Inc". Iphone family device belongs to the category of "Smart Phones" and is considered to be one of its leaders. The operating system running on it is the "iOS" also developed by "Apple Inc" especially for this device. This is an assurance for great compatibility between the operating system and the device. The programming language is used for developing iphone applications on iOS is the "Objective C" language which is familiar to the old time classic "C/C++" programming language. "Objective C" as its name indicates is another object oriented software programming language and is adopted by "Apple Inc" as its main programming language for all its systems running the Apple's operating system MAC OS X and iOS.

# CHAPTER 2

# IPHONE 4G DEVICE

The iphone 4G mobile device is a mobile device in the category of smart phones used by thousands of people every day for communicating through the GSM mobile network. Although, like most of the devices of its category, is provided with a stack of accessory devices like an 5 Mega Pixels resolution camera, accelometer, light detectors, GPS and many other helpful devices. In coin recognition project, only the camera device was used.

## 2.1. Iphone 4G device

The more important part of the iphone 4G mobile device is the Apple A4 chipset it uses. Apple A4 is an all in one chipset and is used in many «Apple Inc» mobile devices like the «iPad», «iPod» and «Apple TV». The A4 System-on-Chip (SoC) is designed by «Apple Inc» but manufactured by «Samsung Electronics». It is based on the ARM processor architecture using an ARM Cortex-A8 CPU running at 1GHz paired with a PowerVR SGX 535 graphic processor (GPU)

| ***Apple A4 Chipset*** | |
|---|---|
|  | |
| Produced | From March 2010 |
| Designed by | Apple Inc. |
| Common manufacturer | Samsung Electronics |
| Max. CPU clock rate | 1 GHz |
| Min. feature size | 45 nm |
| Instruction set | ARM v7 |
| Cores | 1 |
| L1 cache | 32 kB Instruction<br>32 kB Data |
| L2 cache | 512 kB |
| Predecessor | Samsung S5L8920 |
| Successor | Apple A5 |



*Figure 17: Apple A4 Chipset Summary*

The A4 processor package does not contain RAM, but supports package-on-package (PoP) installation. The top package of the A4 usually contains two low-consumption 128 MB DDR SDRAM chips (total 256Mb). Although, for the iPhone 4G, Apple is using two chips of 256 MB for a total of 512 MB RAM. At last, three editions of iphone 4G are available on the market as far as its storage memory offering the capacity of 8, 16 and 32 Gigabytes SSD physical memory.

**Iphone 4G Specification Summary**

| | |
|---|---|
| OS | iOS 4, iOS 5, iOS 6 |
| Chipset | Apple A4 |
| CPU | ARM Cortex-A8, 1GHz |
| RAM | DDR SDRAM 512 Mb (2x256Mb) |
| GPU | PowerVR SGX535 |
| Storage | 8/16/32 Gb SSD |

*Figure 18: Iphone 4G Specifications Summary*

## 2.2. Mobile Operating System iOS

The iphone 4G device like all in the iphone family is using the iOS mobile operation system. The iOS operating system is developed and distributed by “Apple Inc”. It manages the device hardware and provides the technologies required to implement native applications. iOS is derived from MAC OS X, with which it shares the Darwin foundation, and is therefore a Unix operating system.

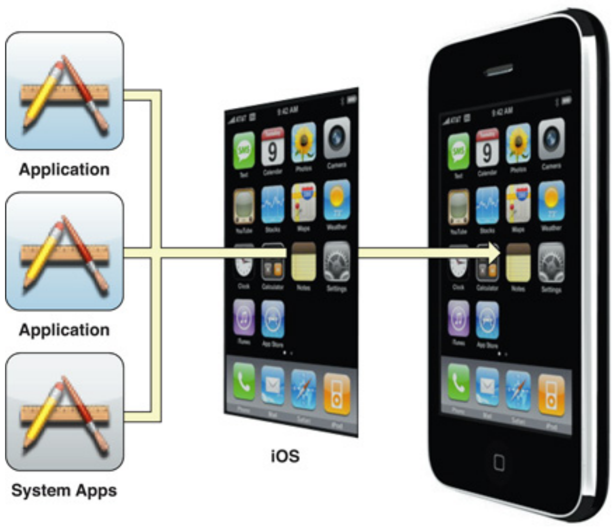



*Figure 19: iOS Applications Layer*

In the iOS architecture layer, at the highest level, iOS acts as an intermediary between the underlying hardware and the applications that appear on the screen. The applications communicate with the hardware through a set of system interfaces that protect the application from hardware changes (Figure 19). This abstraction makes it easy to write applications that work consistently on devices with different hardware capabilities.

In iOS, there are four abstraction layers (Figure 20): the Core OS layer, the Core Services layer, the Media layer, and the Cocoa Touch layer.

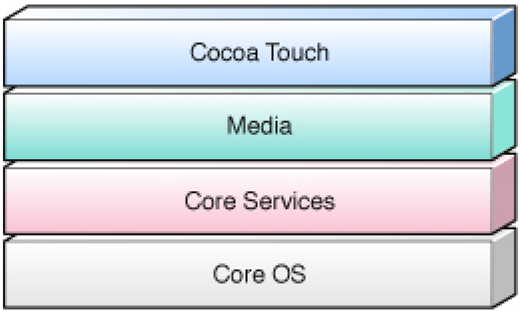



*Figure 20: iOS Layers*

At the lower layers of the system are the fundamental services and technologies on which all applications rely. The Higher-level layers contain more sophisticated services and technologies.

The higher-level frameworks are there to provide object-oriented abstractions for lower-level constructs. These abstractions generally make it much easier to write code because they reduce the amount of code you have to write and encapsulate potentially complex features, such as sockets and threads. Although they abstract out lower-level technologies, they do not mask those technologies. The lower-level frameworks are still available for developers who prefer to use them or who want to use aspects of those frameworks that are not exposed by the higher layers.

# CHAPTER 3

# COIN RECOGNITION PROJECT APPLICATION

The Coin recognition Project iOS Application offers a user the ability of recognizing coins identity by photographing them or grabbing its images from the device's photo album. The recognition comes by matching the image the user provides with a total number of coins images saved in applications database. The application keeps important samples of different kind of coins and tries to detect them inside the user's photo using the normalized cross correlation pattern recognition algorithm. The high correlation results give the suspicious of image matching.

As the application is developed for experimental purposes, it is released with an editable database. This means that the user is free to add or remove coins and its samples from the database. The user can add his own image samples and experiment himself. The application is also designed to be used for any image pattern recognition experiments using the "normalized cross correlation" algorithm and is not limited in coin recognition.

## 3.1. Architecture

The architecture of the «Coin Recognition Project» Application, as object oriented programming asserts, is based on 2 main class libraries. The first one called «Coin Managing» is the one that is used for managing the data the applications uses. These data are the images, the sample images and the information needed by the application. The second one is called «Image processing» and like its name indicates, it is the class used for processing the images. This class is the one that implements the image processing algorithms.

Lots of classes are used in the application for displaying the user graphic interface and supporting these two main classes but this section out of the scope. The next chapters are focused on the application's two main class libraries.

### 3.1.1. Coin Managing Library

The «Coin Managing» library is the one that is responsible for the data management. The application handles three portions of data:

- ***Country List:*** It's the list of countries (categories) that the coins (images) are categorized to. It is just an array list, saved in the device memory. Its data structure can be seen in Figure 21.

| | |
|---|---|
| Data Structure | Country #1 |
| | Country #2 |
| | … |

| | |
|---|---|
| Example | Europian Union |
| | USA |
| | … |

*Figure 21: Application Database – "Country List" Data Structure*

- ***The Coins List:*** This is also an array list saved in memory. Although it is a little more complicated from the «Country List», as each element of the list contains more information. This list is used for listing the coins that are used by the application. The information it contains is the name of every coin, the country it belongs and a record showing if the coin is in active mode or not. The data structure of this dataset can be seen in the following figure (Figure 22).

| | | |
|---|---|---|
| Data Structure | Coin #1 | Name |
| | | Country |
| | | Active |
| | Coin #2 | Name |
| | | Country |
| | | Active |
| | … | … |

| | | |
|---|---|---|
| Example | Coin #1 | 1 Euro |
| | | European Union |
| | | YES |
| | Coin #2 | 1 Dollar |
| | | USA |
| | | NO |
| | … | … |

*Figure 22: Application Database – "Coin List" Data Structure*

- ***The Coin Data:*** For every coin used in the application a data file exists in the iOS user's documents path where its data is saved. The application keeps information about every coin like a pattern image of the coin, its name, its country, and the template image with its samples that the application uses for recognizing it. For the template and each sample image extended information is kept. In Figure 23 detailed information about this data structure is provided. The usability of this class is in loading, updating and saving these kinds of data.

| Data Structure | | |
|---|---|---|
| Name | | |
| Country | | |
| Samples (Array) | Sample #1 | Width |
| | | Height |
| | | Deviance |

| | | |
|---|---|---|
| | | Mean |
| | | Data |
| | | Correlation |
| | Sample #2 | … |
| Image | Width | |
| | Height | |
| | Image Data | |
| Template Image | Width | |
| | Height | |
| | Image Data | |

| Example | | |
|---|---|---|
| 1 Euro Coin | | |
| European Union | | |
| Samples (Array) | Sample #1 | 20 |
| | | 20 |
| | | 234.67 |
| | | 146,53 |
| | | Pixel Array |
| | | (Pixel – Mean) Array |
| | Sample #2 | … |
| Image | 150 | |
| | 150 | |
| | Pixel Array | |
| Template Image | 160 | |
| | 160 | |
| | Pixel Array | |

*Figure 23: Application Database – "Coin Data" Data Structure*

All data structures uses "XML" format (OS X & iOS ".plist" files) for saving its contents.

As mentioned in order to make the algorithm run faster is that each time the user adds a sample in a coin, the application calculates its mean, its deviation and its correlation array (correlation[i] = pixel[i] – mean). This information is kept in the relative indexes in the data structure shown in Figure 23.

### 3.1.2. Image Processing Library

The «Image Processing» library is the one that is used in order to apply the «Normalized cross correlation» algorithm on the images and produce the wanted results in order to be projected to the user. In addition it offers a useful set of functions for cropping, resizing, rotating, converting RGB colored images to grayscale and 24 or 32 bits-per-pixel images to 8 bits-per-pixel ones. The function is used for converting colored images to grayscale is the following,

$$F(x) = (x_R + x_G + x_B)/3$$

The «Normalized cross correlation» algorithm function was referred in chapter 1.1. It is worth to refer again that the deviation, the mean and the correlation (correlation = pixel[] – mean) values of each sample image is correlated with user's main image is pre-calculated by the time it was added in the database. This technique reduces the amount of complexity and the period of time the algorithm needs to be executed.

The recognition process was described in chapter 1.3.6. In these chapter the implementation of the «Normalized Cross Correlation» algorithm used in the core of the process is described by the form of pseudo code follows in .

```
Image image_tile;
List samples_List;
```

the coin's image tile for correlation
List of the coin's corresponding tiles in data format as described in chapter 2.1.1

```
for ( i=0; i<image_tile.size; i++) {
tile_mean += image_tile[i];
}
tile_mean /= image_tile.size;
```

$$M = \frac{\sum_{i=0}^{i=N} x_i}{N}$$

```
for ( i=0; i<image_tile.size; i++)  {
tile_correlation[i] = (image_tile[i] – tile_mean);
tile_deviance = tile_correlation[i]* tile_correlation[i];
}
```

$$C_i = \left(x_i - M\right)$$

$$\sigma^2 = \sum_{\iota=0}^{\iota=\mathrm{N}} (\chi_\iota - \mathrm{M})^2 = C^2$$

```
for( i=0; i<sample_list.size; i++) {
for ( j=0; j<image_tile.size; j++) {
correlation[j] +=
tile_correlatio[j]*sample_list[i].sample_correlation[j];
} }
correlation[i] /=
sqrt(tile_deviance*sample_list[i].deviance);
```

$$NCC = \frac{\sum_{i=0}^{i=N} C_{i^{tile}} \bullet C_{i^{sample}}}{\sqrt{\sigma^2{}_{tile} \bullet \sigma^2{}_{sample}}}$$

```
correlation [i] = (1 + correlation [i]) * 50;
```

$$NCC = \frac{1 + NCC}{2} \bullet 100$$

Converts result values from [-1,1] to [0,1]

*Figure 24: Image Processing Library – Normalized Cross Correlation Algorithm*

## 3.2. Application User Graphic Interface

By tapping the “Coin Recognition” Application Icon in the iOS Spring Board, the Main View (Figure 25) will be shown. This view has three buttons.

- ***Camera button:*** By pressing this button the user is going to be forward to the Camera or Photo Album layer in order to choose a photograph from the device photo album or to capture one with the device’s camera. This photograph is the one that will be used for recognition.

- ***Recognize Button:*** This button is inactive by the time the user starts the application. That’s because no photograph has been chosen. The «Recognize» button is the one that starts the recognition process. After the user has made a choice using the «Camera» button, the «Recognize» button is activated.


*Figure 25: Application User Interface - Main View*

- ***Settings Button:*** The «Settings» button leads to “Settings” menu view (Figure 28) of the application. In this section of the application the user has the ability to configure the application and manage its database.

### 3.2.1. Image Picker Menu

Every time an image is needed for the application functions flow the image picker menu appears. The image picker menu is a modification of iOS camera application menu as shown in Figure 26 an optional gold circle appears in the center of the camera in order to lead the user to capture a useful image. This circle appears if the user has enabled the «auto crop» tool that crops the image inside the circle by default in order to use it for the recognition process. If the user decides to crop the image himself the image edit view will appear as long as he captures an image. Every image captured is automatically saved in the device’s camera photo album. The user can tap the «Photo Album» button in order to choose from an already saved image. When picking an image from the photo library, the user is also prompt to edit it in order only the coin view to be visible as shown in Figure 27.


*Figure 26: Application User Interface – Image Picker View*


*Figure 27: Application User Interface – Photo Edit View*

### 3.2.2. Settings Menu

«Settings» menu (Figure 28) gives the ability to the user to configure the application on his needs. At the top of the view there are three submenus for configuring the application.

- ***Coins Settings:*** In this submenu the user can add new coins or delete other ones from the application database. These coins are the pattern for recognizing coins.
- ***Active Coins:*** «Active Coins» submenu lets the user choose which coins will be used (activated) in the recognition process.
- ***Country Settings:*** In this submenu, the user can create and delete countries. The «Country» information is used like a category so the user can organize better the coins in the database.

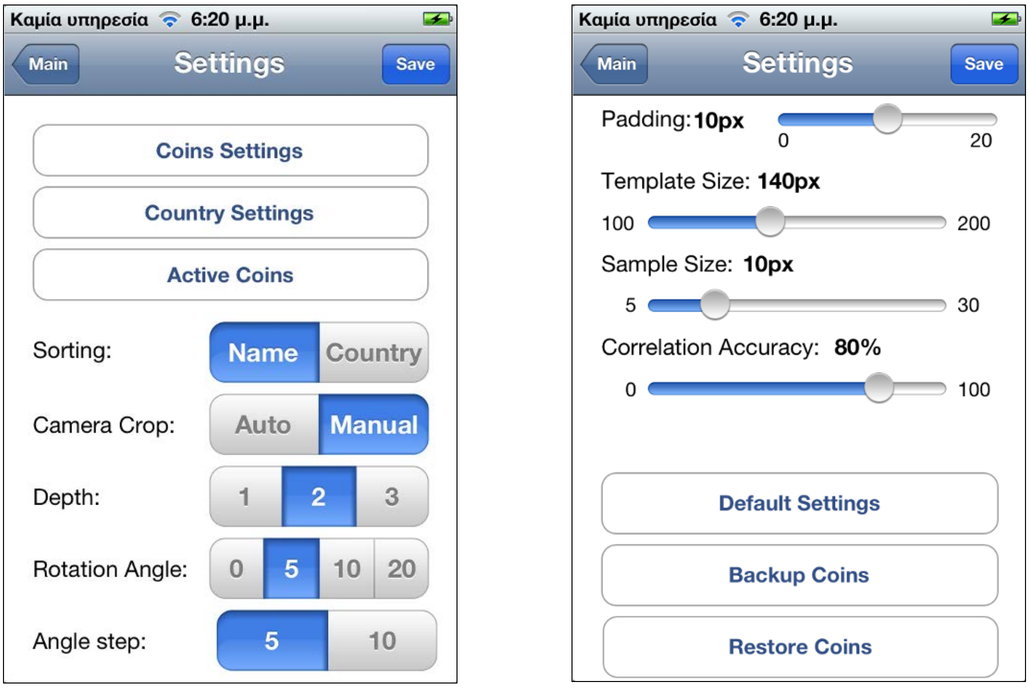



*Figure 28: Application User Interface - Settings View*

By the middle height of the «Setting» Menu view there is a section of some useful configuration options for the application.

- ***Sorting:*** The sorting option specifies whether the coins will be display sorted by its name or grouped and sorted by its country every time the coin's list appears.

- ***Camera Crop:*** The user has the ability to choose whether he will edit the images he capture by himself or the application will. When the «Auto» option is chosen a gold circle leads the user to grab the coin inside it (See Figure 26). If the «Manual» option is chosen the edit photo menu will appear right after the image capture.

- ***Depth:*** This option defines the sample depth. A depth of value «1» means that only the center sample will be used for the correlation. A value of «2» means that all the neighbor sample sized areas of the center sample can also be available for correlation if it is needed.

- ***Rotation Angle:*** when an image is captured it is obvious that rotation mismatches may exist. The «Angle Check» option defines the value of the rotation angle that the user wants the application to check. If for example the $10^{o}$ degrees option is chosen, the application will also correlate the coin images with at last $\pm 10^{o}$ degrees rotated instances of the coins templates.

- ***Angle Step:*** The angle step option defines the rotation step until the «Angle Check» limit is reached. So, if for example the user sets the «Angle Step» option «5» and the «Angle Check» as 20, the coin image will be correlated with 0 $^{o}$, ±5 $^{o}$, ±10 $^{o}$, ±15 $^{o}$ and ±20$^{o}$ rotated instances of the coins templates.

- ***Padding:*** Another mismatch that is also certain to occur is the coin slide around the center of the image. «Padding» option defines this possible mismatch radius around the center. If for example the value «10» is chosen for this option, the application will correlate every sample of a template in a 10 pixel greater in all direction area.

- ***Template Size:*** This option defines the template image size to be used for the correlation

- ***Sample Size:*** This option does the same thing with the previous one, this time defining the sample images size. It is easily tangible that the Sample Size multiplied with the Depth cannot exceed the Template Size

- ***Accuracy:*** This is the parameter that defines the minimum percentage distance from the maximum correlation coin result that the rest coins have to succeed in order to qualify to the next tile correlation (Chapter 1.3 – Pick over stage). If for example the «Accuracy» parameter is defined as 80 all coins that succeed a correlation value greater than the 80% of the maximum one will be correlated with the next tile. This mean that all the coins that has achieved a less than 80% correlation result will not be used in the next stage. As highest these parameter is, more coins are about to be disqualified in each tile correlation. This will make the process sooner. On the other hand a very high value of «Accuracy» parameter may lead to inaccurate results.

At the bottom of the «Setting» menu there are three auxiliary buttons for restoring the default application's settings values, backing up the coins data or restoring them.

### 3.2.3. Coins Settings Menu

As mentioned, in the «Coins Settings» menu the user can add, edit and delete coins from the database. The main view (Figure 29) of this submenu displays a list of the coins saved in the database.


*Figure 29: Application User Interface - "Coins Setting" View (Name Sorting, Country Sorting)*

In the upper right corner of the task bar there is a «plus» button. By pressing this button, the «Add New Coin» (Figure 30) task starts. The user has to fill the text field with the name of the coin for sure. There more information to be filled but it is not obligated, like the Country (Figure 31) the coin belongs, a pattern image and the template image of which the samples will be extracted.

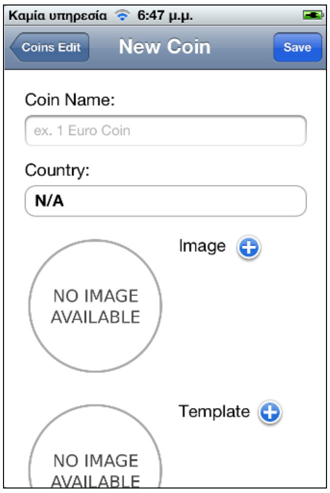



*Figure 30: Application User Interface - "New Coin" View*

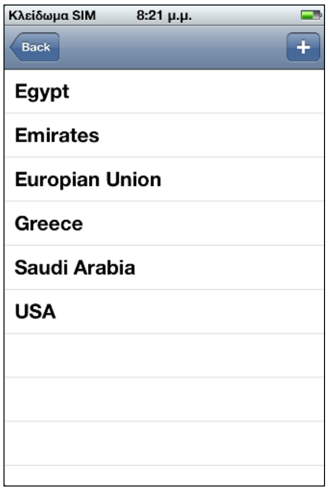



*Figure 31: Application User Interface - "Chose Country" View*

Selecting a coin from the list will forward to the «Coin Edit» view (Figure 32). In this view the user can rename the coin, change its country, its image and template image, or either delete the coin by pressing the red «Delete Coin» button in the bottom of the window.

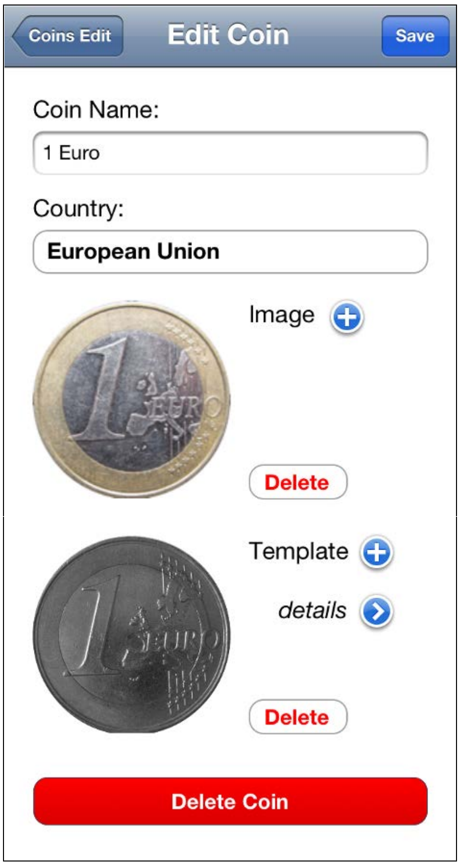



Figure 32: Application User Interface - "Edit Coin" View

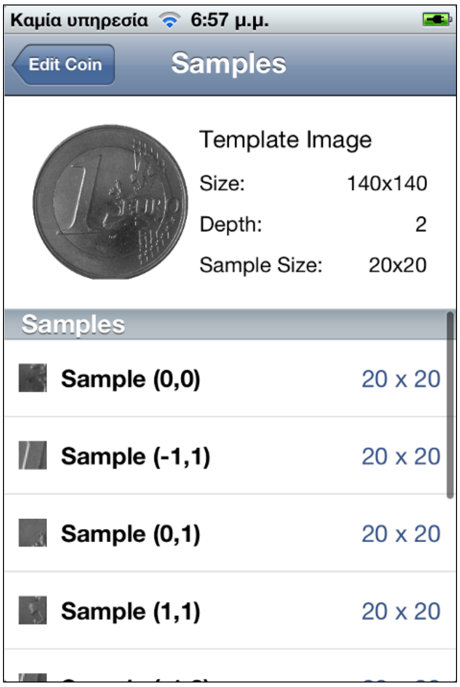



Figure 33: Application User Interface - "Samples" View

The blue indicator button right of the «details» field leads to the «Samples» view (Figure 33). In this window the user can see the images of the coin's samples.

### 3.2.4. Country Settings Menu

In this submenu the user can simple add or delete countries (or categories). Pressing the «plus» button is for adding new countries (Figure 35) and sliding over a country presents the «Delete» button to delete the country from the list (Figure 34).

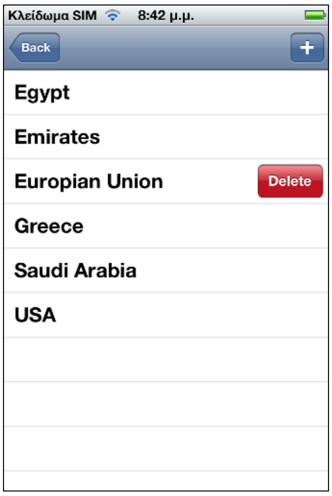



*Figure 34: Application User Interface - "Country Settings" View*


*Figure 35: Application User Interface - "New Country" Popup Window*

### 3.2.5. Coins Activation Menu

«Active Coins» Submenu (Figure 36) is the sector where the user can choose which coins will be used in the recognition process by the algorithm and which ones will stay inactive for the application. A simple two segment «on/off» button is used by the user graphic interface for each coin.

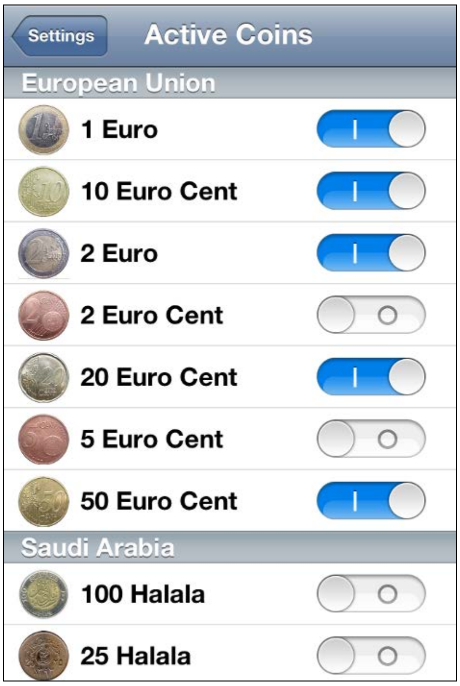



*Figure 36: Application User Interface - "Coins Activation" View*

### 3.2.6. Coin Recognition

Assuming the user has chosen a coin image from the device photo library, the recognition process begins by the time of pressing the «Recognize» button in right bottom corner of the application main view (Figure 37).

By the end of the recognition process the application presents the details of the recognized coin as shown in *Figure* 38. The user can access the details of the recognized image as in the «Coins Edit» Menu (Figure 32) by pressing the «Coin Info» button at the bottom of the view.


*Figure 37: Application User Interface - "Main" View While Recognizing Process*


*Figure 38: Application User Interface - "Correlation Results" View*

At last, by pressing the «Recognition Stages» button the application represents the details of the recognition process grouped by stages as shown in Figure 39. Details about the time last each stage and the correlation results succeeded by each template are shown. At the «Finalists Stage» section are presenting the templates that manage to reach until the last tile where the mean correlation value is the one that defines the recognized coin.

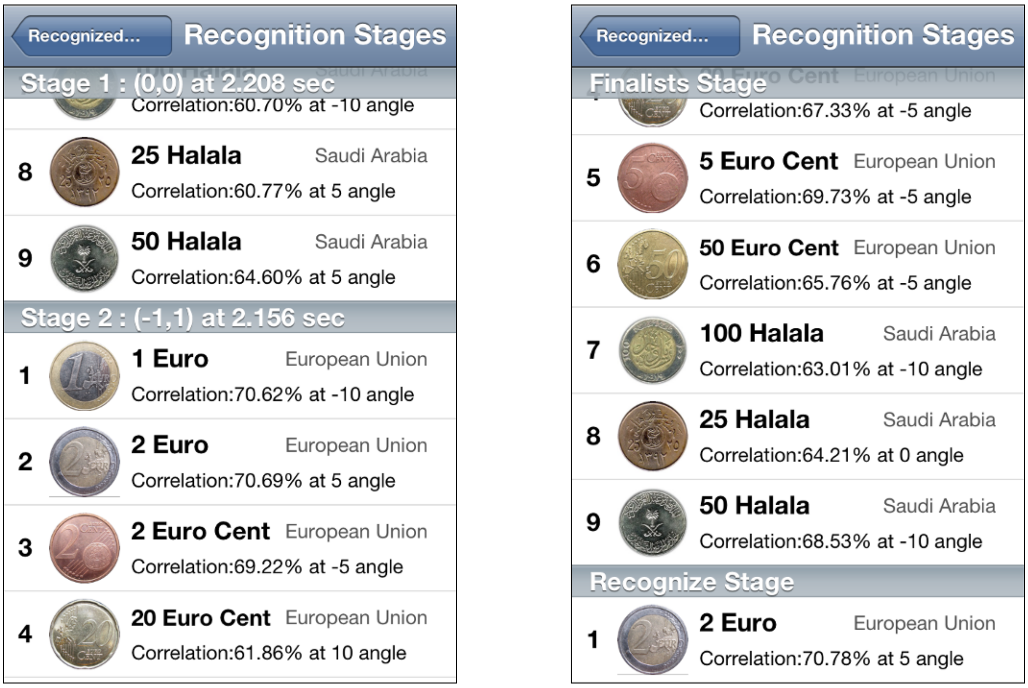



*Figure 39: Application User Interface - "Recognition Stages" View*

The recognition process duration depends on lots of factors. High resolution images, a lot of active coins in application database and many tiles used for recognizing the coin are factor that rises the recognition process duration. In Figure 40 table the effect of each parameter of the application to recognition process is presenting. The «Depth» Parameter as mention in the figure is depending on the «Accuracy» parameter. This is reasonable as if the «Accuracy» parameter is high more coins will be disqualify in every stage of the recognition process and may not all tiles need to be used. For example in the maximum value of «Accuracy» (100%) only the main tile (0,0) will be used independent of the depth parameter. On the other hand setting the «Accuracy» parameter to 0% it is like disabling the disqualification process using all the templates in all stages and recognizing the coin by the mean correlation values.

| Parameters | Min | Max | Accuracy (Max) | Defaults |
| --- | --- | --- | --- | --- |
| Active Coins | Speed up | Slow down | No effect | 10 |
| Depth | Possible speed up (Depends on accuracy) | Possible slow down (Depends on accuracy) | Increase | 2 (9 tiles) |
| Rotation Angle | Speed up | Slow down | Increase | 10 (±10) |
| Angle Step | Slow down | Speed up | Increase | 5 |
| Padding | Speed up | Slow down | Increase | 10 |
| Template Size | No effect | No effect | Increase (Depends on the Sample Size) | 140 |
| Sample Size | Speed up | Slow down | Increase | 20 |
| Accuracy | Slow down | Speed up | Decrease | 80% |

*Figure 40: Coin Recognition Process – Parameters Speed Affect*

Trying to create a model of time of the process without using the Accuracy parameter speedup that has unpredictable effect in the process as it is depending on the similarity between the coins stored in the database, we can conclude to the following theoretical function,

$$T = Coins \cdot Depth \cdot \left( \frac{2 \cdot Rotation}{AngleStep} + 1 \right) \cdot (2 \cdot Padding + 1)^2 \cdot \frac{Sample}{Sample_{MIN}} \cdot T_{MIN}$$

$Sample_{MIN}$ variable represents the minimum sample value and the $T_{MIN}$ one the time needed for correlating that sample with the minimum values of the rest variables.

As it is understandable the «Padding» parameter is the one that affects the most the execution time than the rest parameters. This can be noticed in the results of the Figure 39. Figure 41 represent the testing results on the iphone 4 mobile devices.

| Accuracy | Coins | Depth | Sample Size | Template Size | Padding | Angle Step | Rotation | Time (sec) |
|---|---|---|---|---|---|---|---|---|
| **0%** | 10 | 9 | 20 | 140 | 10 | 5 | 10 | 18,9 |
| **0%** | 10 | 9 | 20 | 140 | 10 | 5 | 5 | 11,3 |
| **0%** | 10 | 9 | 20 | 140 | 0 | 5 | 10 | 0,68 |
| **0%** | 10 | 9 | 20 | 140 | 10 | 5 | 0 | 4,00 |
| **0%** | 10 | 9 | 20 | 140 | 0 | 5 | 0 | 0,14 |
| **0%** | 10 | 9 | 20 | 100 | 10 | 5 | 0 | 3,90 |
| **0%** | 10 | 9 | 20 | 140 | 0 | 5 | 20 | 1,30 |
| **0%** | 10 | 9 | 10 | 100 | 0 | 5 | 20 | 0,77 |
| **0%** | 10 | 9 | 10 | 100 | 0 | 5 | 0 | 0,09 |
| **0%** | 10 | 9 | 10 | 100 | 20 | 5 | 0 | 11,4 |
| **0%** | 10 | 9 | 10 | 200 | 0 | 5 | 0 | 0,10 |
| **0%** | 10 | 9 | 30 | 200 | 0 | 5 | 0 | 0,20 |
| **0%** | 10 | 9 | 30 | 200 | 0 | 5 | 20 | 2,59 |

*Figure 41: Coin Recognition Process – Times Table*

# CONCLUSIONS

As mentioned before, the “Coin Recognition Project” is a research project on image pattern recognition area. The iOS application developed is bases to be used for further more research. There is a large range for evolvement in all aspects.

Today, mobile devices are becoming more and more powerful. The memory and the cpu capabilities reach the range of the desktop computers we used to use some years ago. The image processing technology is also improving itself time to time. This is a very positive pattern for the future of mobile world. The Apple iphone 4G mobile device respond very weill to the demands of the project, even if speedup techniques were set. This fact establishes that the time that mobile devices will reach the standards needed for the use of image processing technology is not far away.

As far as the iOS application developed, upgrades can be implemented for speedup improvements. Others custom techniques can be used to reduce the amount of time needed or accelerate the code. Combinations of different image processing algorithms can be used for straightforwardness. Edge detection algorithms are usually proposed to be applied before normalized cross correlation one is used. All these issues are to be discussed and implemented as the next step forward.