DEGREE THESIS

Embedding and decoding hidden data channels on computer displays

Lehrstuhl für informationstechnik Schwerpunkt komunikationselectronik (LIKE)

Friedrich-Alexander-Universität Erlangen Nürnberg

ETSI de Telecomunicacions

Universitat Politècnica de València

In collaboration with

Fraunhofer Institute for Integrated Circuits IIS

Degree:	Telecomm & Electronic. Eng.
Student:	Antonio Juan Querol Giner
Matriculation number:	21427894
Supervisors in Fraunhofer IIS:	Dipl.-Ing. Stefan Krägeloh,
	Dipl.-Ing Jörg Pickel
Professor:	Prof. Dr.-Ing. Heinz Gerhäuser
Start date:	15.03.09
Closing date:	15.09.09

Declaration

I assure that this Degree thesis is done independently, without alien help and without the employment of other sources and auxiliary material which are not explicitly citate on this degree thesis

Antonio Juan Querol Giner

Abstract

Title

Embedding and decoding hidden data channels on computer displays.

Abstract

In a given setup, a LCD screen is mounted horizontally so that objects, e.g. game tokens, can be placed on the surface of the display. A PC displays information on the screen, e.g. a floor plan of a board game. The object placed on the screen is equipped with one or more optical sensors that "look" at the display area under the object. The information captured by the sensor is transmitted back to the PC that displays the information (after optional pre-processing the object). In order to determine de position of the objects in the screen, the screen is partitioned into a large number of separate areas. A separate hidden data channel has to be implemented in each of these areas. The data transmitted through these data channels can be used to transmit position identifying the given area. The task of this work is to research and design a system for implementing such a multitude of hidden data channels on a video display with the following constraints:

- The displayed hidden data channel shall be invisible or at least unobtrusive to the human eye in the chosen setup.
- The capacity of the channel should be as large as possible.
- The system should be able to function if the sensor is placed between two, or three of the hidden data channel areas.
- The system needs to be robust against typical error signals like "strobing back light of the display".
- The system should be able to adapt to special conditions found on different types of display families.

Keywords

LCD, hidden channels, localization

Dedication

I want to dedicate this degree thesis, first of all, to my parents Antonio and Cinta. Because without their help I cannot finish my carrier and arrive to this point of my life. I am also very Grateful to my Sister Maria Consuelo, for her advices and suggestions during all my life. And, why not, also say thanks for all the discussions that we had in the childhood like those will have my nephews Rodrigo And Alejandro in the future. Be careful Manolo!. And Finally I would remember my grandmother Cinta

I want to write another special words to my girlfriend Raquel, for her compression, but really, all the things that I should write here are two long for this few words, so I will say in the shortest way: I love You Raquel.

Maybe one man is remembered by his conquests, but always is better remembered by his friends. So now, I want to say thanks to my friends in my born town Benicarlo, especially to David. During my studies in Valencia I made a lot of friends that really helped me, the list is too long to write here but there are some names that must be here. Victor, my partner in the easy labs and also in the difficult ones. Nacho and Maria, I think that these two are the smoking friends which I spend more time studying in the school. My lungs also want to say thanks. The last sentence is ironic. The last group is named only by their nick names, Miguelón and Jodarin, thanks. Please, all the other friends that are not cited in this dedication and they think that should be here do not be angry with me, when you see me in Spain remember me this and I will invite you to something for drinking.

I'm also grateful to my teachers from the school, secondary school and the University for their explanations and advices.

And here in Erlangen the last step in my life, thanks to Gracia for make all easier, And to my supervisors Stefan, Mercé and Juliane for their confidence in me to make this degree thesis in Fraunhofer IIS.

Finally I do not know why, but I'm feeling that I must say thanks to my aunt Maria.

Contents

List of figures

List of abbreviations

3-CCD………………..	three separate Charge-Coupled Device
ADC…………………..	Analog to Digital Converter
AWGN………………	Additive White Gaussian Noise
BER…………………….	Bit Error Rate
BPSK………………….	Binary Phase Keying
CFA……………………	Colour Filter Array
CPU……………………	Control Proceed Unit
DEP……………………	Density Spectral Power
FFT…………………….	Fast Fourier Transform
FPGA…………………	Field Programmable Gate Arrays
IP……………………….	Internet Protocol
ISI………………………	Inter Symbol Interference
LCD…………………….	Liquid Crystal Display
MIMO………………..	Multiple Input Multiple Output
OA………………………	Operational Amplifier
OSI…………………….	Open System Interconnection
PAM…………………..	Pulse Amplitude Modulation
PC………………………	Personal Computer
PIC…………………….	Peripheral Interface Controller

RFID……………………	Radio Frequency IDentification
RGB…………………….	Red Green Blue
SNR…………………….	Signal to Noise Ratio
TCP……………………..	Transmission Control Protocol,
TUI……………………..	Tangible User Interface
USB…………………….	Universal Serial BUS

Chapter 1

Introduction

1.1 Introduction

In this degree thesis, I'm trying to show how to build and hidden data channel, it has to be hidden for the human eye, So first of all, we are going to see the display characteristics of a LCD screen and the human perception of visual signals. But I encourage the following engineer that work in this theme to make a more accurate study on this fields, in order to make a system that can be in the market and not cause any problem to the human eye. For the study of the LCD, we are going to make some experiments on the display that is going to be used (SAMSUNG SYNC master 193T). And look for some information on the Samsung Web.

For the human response to our system, we need to see the response that has the people to system and look for old studies in this area [45]. In order to achieve the first topic, I try to make a little test to my partners in Fraunhofer IIS more details will be written later.

According to these things we have to build the channel. So in this thesis, I will try to describe the modulator (transmitter) of the signal that we are going to send to the screen, and the demodulator (receiver). Finally the channel is modelled like one AWGN channel.

In this point, it will be good to show one figure that explains our situation. In the screen we send the signal. The part that I will build and has to send the signal to the monitor is our modulator, and it has no physical implementation except that the wire that carry the signal to the LCD display. This part has to divide the screen into different areas, and code the information of the area's position.

Then the token, placed on the screen, will capture this signal. And then will be the time for the demodulator to get and make some processing on the signal, in order to find the position coded in the information that the area, where the token is located, is sending.

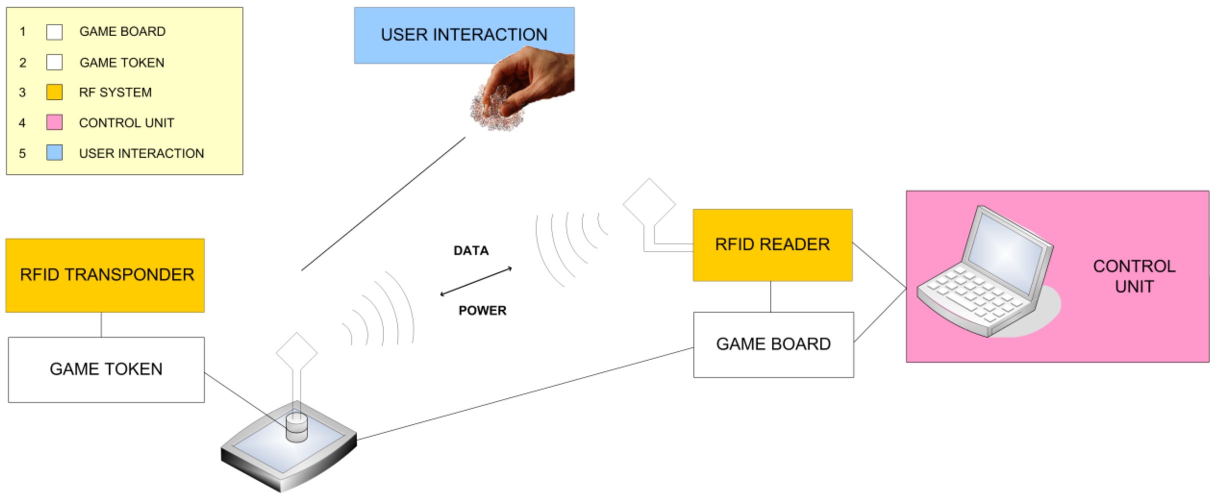

Fig. 1.1 Monitor and token

The diagram of the system is the well known diagram for a digital communication system. Likes show the figure 1.2.

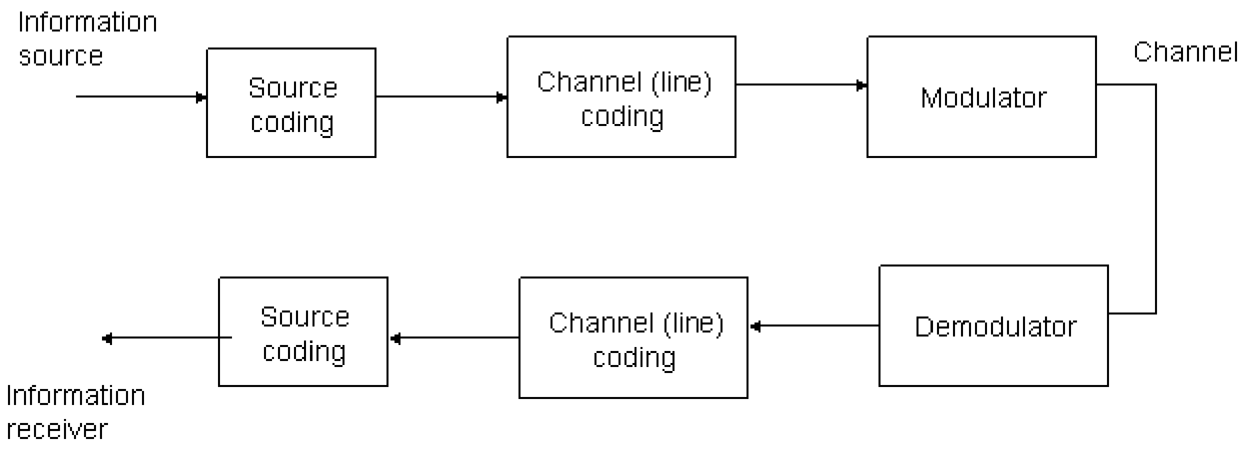

Fig. 1.2 Simple Digital communication system

The information source is for us the position of the divisions that we make in the screen, then this information is coded, and we send to the screen, this wired channel, for us, is a perfect one and we suppose that has no errors. So really this modulator is only software that gets the position and code it, to brightness levels in the screen, for example. This part is all made in Java and a detailed description will appear in his chapter.

The channel and his noise is the joint of the problems that has the screen displaying the signal (inclusive a little memory channel) and the some noise that is added to our signal, this noise can be external light sources (sun, lamps, …), and finally some AWGN.

The demodulator will be an all-digital receiver, because the sensor it's a light sensor and has an ADC placed near this sensor, later, I will write a little description of this, but a detailed description of this is written in [1]. When we have the signal in the token we are going to make the normal processing in a demodulator, to get the sent information. For the moment, I want to say that this part has the physical part of the sensor and one OA. This thing is connected to the USB-Audio capture. And this sampled data is read by the computer using Matlab. Finally the version of the demodulator is all build in Matlab. This process will be explained in the correspondent chapter.

This architecture has to be implemented embedded the game development, in order to have a good response of the whole system.

1.2 Motivation

Looking at the introduction our system is like a communication channel, which we have to model and control. The task is to transmit the information of the position as fast as possible; this will be the hardest problem of the system because we are limited by the refresh rate of the screen (in our case 60Hz). But for the future we have to break this barrier, In the future work I will explain the methods for this. One is put then information in the three colours of the monitor (RGB system), for the moment we only work with brightness levels.

We have to make some experiments for modelling the channel, and find the correct form for sending data, so I will try to find the ideal modulation for this channel, and the frame data that avoid more errors in the channel and has the minimum length for a faster transmission.

Another task is the implementation of the modulator and demodulator, for this I need to remember and read a lot of theory of signal transmitters and receivers. But this is a good goal to be sure that the concepts that I learned in my home university works in the real life design.

In this thesis we have the problems that have the normal communication system and the problems that the game design has, so we have to look closer to the integration of these two topics in the global system.

Finally the project all together is a very big one, it is impossible to build by only one engineer, so it's important team-work to arrive at the correct solution. In this case it is important to work with the person that makes the acquisition toolbox of the token. Because in this thesis, I'm going to work with the signal that the token detects.

1.3 Goals

For a good definition of the goals, in the beginning I will describe the thesis goals like it was only a single project and then we have to make the integration of this in the game architecture.

The most important goal on this topic is to find the position of one token that is placed on one screen. In order to achieve this goal, we can only use embedded video signal in the screen. For the beginning the image displayed in the token will be a static one, but it can also have little moving parts, without affecting the correct signal transfer. Later implementations will work in a non-static image (video).

This detection has to be made as fast as possible because it has to be implement in the control routine of the game, and if we want to make a playable game, the players cannot spend a lot of time waiting that the computer says, in which position are they tokens, and the next instructions for this positions.

This is one of the most difficult problems to solve, because our transmission rate is limited by the refresh rate on the screen, and we are also limited to use only baseband signals for the solution. Finally the repertory of baseband modulations is sufficient for finding one good for us, however these modulations has to accomplish to things, one is to transmit data at high velocity but it's important to be invisible to the human eye. This topic will be written in the modulator part.

As I said before, for the player the signal has to be invisible for playing, because a flicker screen will be very uncomfortable, for a nice playing with the game. This is the second problem to solve. So for the correct solution of the project we cannot think in this problems separately, because they have more relation that at first sight we can find.

According to theme of finding the token in the screen, another important topic is the question what have to make if the token is in the middle of two areas? The translation of this topic for an engineer is what we must do if we have two signals contributions. In this case the solution is using code words that when we add them the result will not be a third word. So this is will create another reduction of available codes for our system. Instead of normal codes, we must use orthogonal codes which satisfy this goal.

Another task is that our system has to adapt to a lot of different families of computer displays, this problem is quite easy to achieve because we use the Java platform.

Chapter 2

Tangible user interface, description

This thesis is placed in the Tangible user interface from now TUI, this project has to build some way of communication between one computer display and some tokens which the user move through the screen. As the reader can see in the picture.

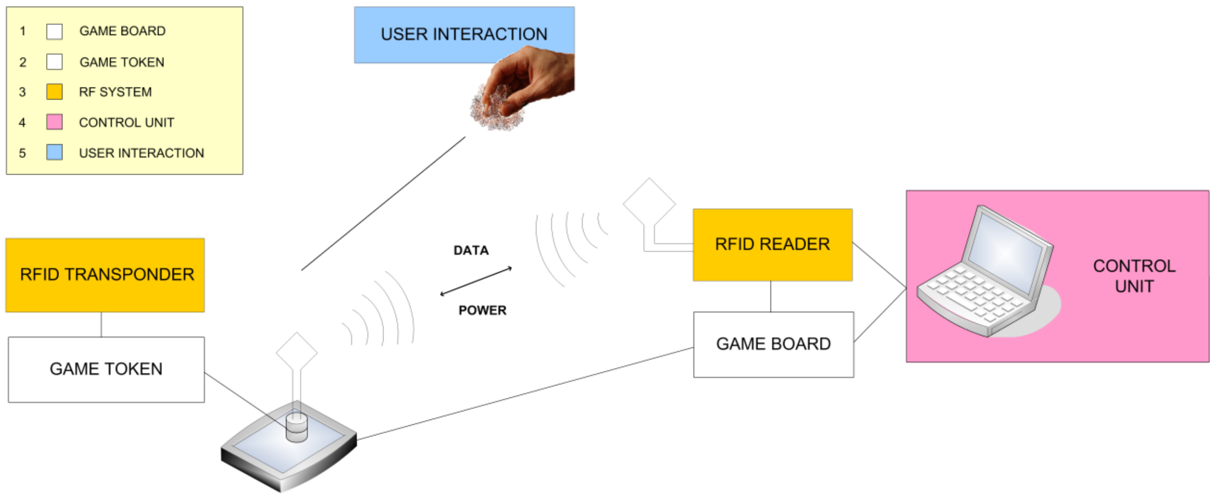

Fig. 2.1 description of components

So the main elements that we have to control for this project are the screen which in this project is the thing that sends the signal to the token. The signal reception for the token.

2.1 connections

The connection between the display and the computer is the normal connection for a pc. So in this case the only thing that we must build is the signal to send to the LCD. But not only for this we have to allocate this signal in the screen with the acknowledge of two things invisibility for the human user and good transmission for accomplishing our goals.

The second channel here is the connection between the screen and the token. This channel is the capital channel in this thesis. The token is able to read the signal on the screen because it has a six-element photodiode array in front of the display. This display looks directly at the LCD. And is able of reading the changes of luminance that has the computer display. In this picture the lector can see the screen a normal pc LCD and the token.

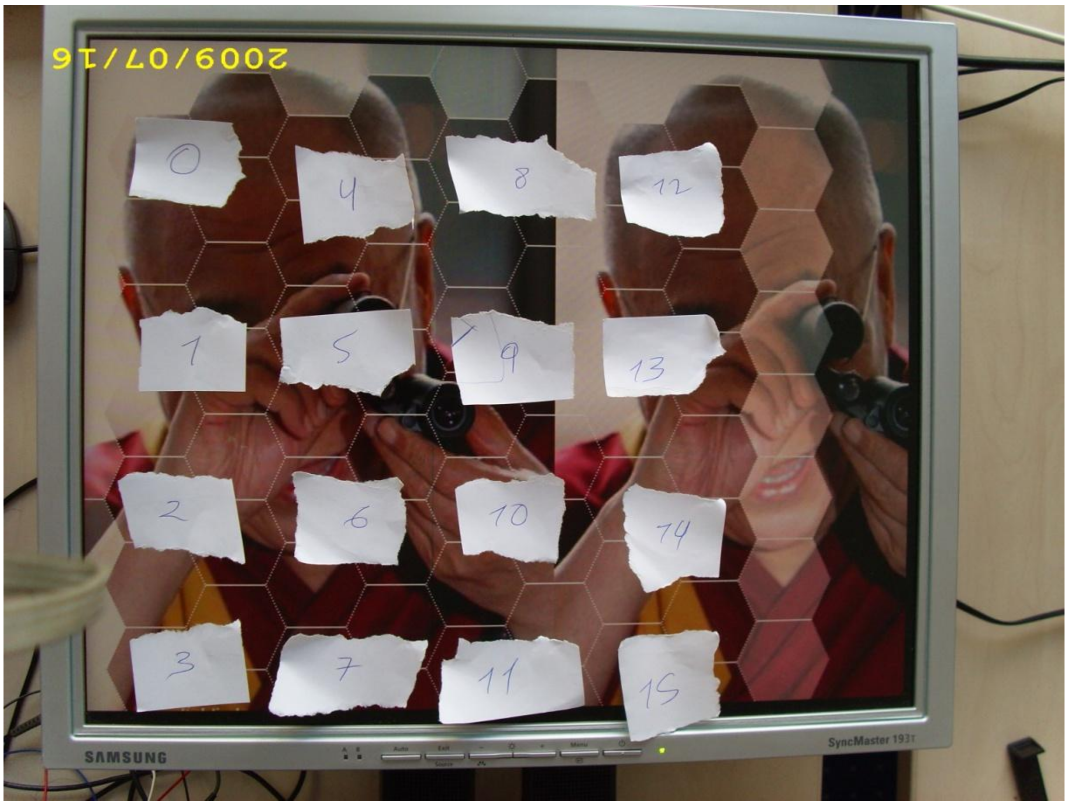

Fig. 2.2 LCD display showing a partitioned image

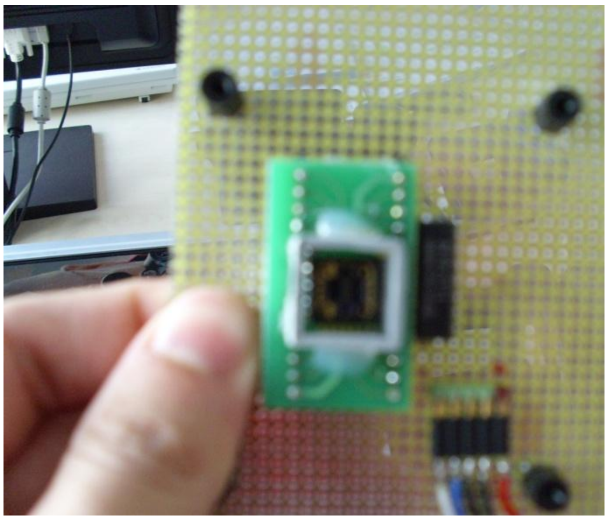

Fig. 2.3 token sensor

Immediately connected to the token is the operational amplifier which amplify the signal.

For the moment I can only use one of the six sensors that has the photodiode. But this is not a problem for a correct communication between the token and the LCD. But in the future the token will be able to read all the 6 sensor and work as a Multiple Input Multiple Output (MIMO). The token will be like a smart antenna and with the addition of beamforming processing for these signals we can get the advantages of this systems for the correct location of the token in the screen, and will be easy to solve the problem of the token placed in the middle using this technologies.

Another task of this channel is sampling the signal for the digital processing, so in this case I will recommend the allocation of one anti-aliasing filter before the ADC. Or make a little of oversampling in order to be sure of taking samples at the Nyquist rate as minimum. On the demodulator chapter I will explain it in more detail.

Finally, the last channel in this project is the connection between the token and the computer. The mission of this channel is to transmit the signal to the computer or maybe the position if in the token we can program the demodulator, in this moment this is a little far away of our position, but maybe in the future.

The chosen protocol for this task is RFID; in my opinion is a very good selection because, obviously, we cannot put a wire in every token to connect them to the computer. So we have to use wireless technologies. The study, of which wireless protocol is better for this case, is far away of this thesis. The only requirement is that the transmitted signal carries the information read by sensor.

2.2 OSI vs. TUI

In the next picture we can see how our system should work compared to the seven layer OSI reference system. For our system we do not use the seven layers but really this not a problem of organization or another thing. To demonstrated that the lector only have to think in the Internet model, also called TCP/IP model. For the readers which do not know this I would say that this system only has four of the seven layers and everybody knows that internet is a good way for communication between people.

In our case the application layer interacts with the user, in three different ways, one is showing the game board to the user, in this board we must embedded the invisible signal, the second one is the information of the game, when the positions of the tokens are calculated, the game have to give new instructions to the players. And finally is that the user can play with the token a really move them.

The next layers are transparent for the user but they are necessary for a good development of the game.

In the Session layer we have to divide the screen in the areas that we need, for example game boxes, limits, different scenarios … And then assign one unique code which defines without error this cell. So we can distinguish this box from the others. Then when we have this, we must build the frame that supports this information. At this point the thing that we make is to code this logical information to light information in the pixels of the screen.

In the data link layer, we add the synchronism and security to our frame. Then we do not make anything more to our signal. We only wait for the signal acquisition after the screen displayed the signal.

The only thing that we must control here is that the transmitted signal is invisible for the human eye.

This part that I have already described is our modulator, this is all made in java and a further description will be done in the correspondent chapter. This part is only placed in the computer so it will be only software for me. In order to achieve that this system must be multiplatform, the programming language is Java.

OSI	PC	Screen	Token	PC
Application	Game	Game Board	--------------------	Game
Presentation		---------------------	---------------------	
Session	Cell assignment	---------------------	---------------------	Cell decode
Transport		---------------------	---------------------	
Net		-------- ----------	Cell decode	
Data link	Walsh, sync…	-------- ----------		
Physical	wire	EO O	OE RFID	RFID

Fig. 2.4 OSI vs. TUI

The next item is the LCD display, for us is the most difficult to control, maybe in the future, we must program the modulator with a low level language that has more control on the screen features.

For this part the control that we have is the value of the RGB that each pixel has to show, when the screen changes, the refresh rate. Really the description here that makes the screen is the transformation of the Electrical signal from de control unit to an optical signal, this signal is the sum of the game board and the embedded signal that divide the board, in many areas. The last signal is the most important in this topic, but to make a good occultation of this signal, we must take a closer look to the displayed game board.

The next part is the token, the token is already constructed, and so for us is only a software modem, which has the three main functions. First of all capturing the signal, then process it to get the data, and finally transmit this information to the PC.

So, in a telematic way we can think on it, like it was only a simple router that has the function of the translate protocols, from the screen to the pc, i.e. capture the signal and put it in RFID frame for transmit it to the computer. And can also work with the frames of the received signal in order to extract the coded information.

Now the token is only able to get the signal and amplify it, so the process for demodulation is made in the computer. It has to get the frames and decode it in order to extract the information. In the data link layer, it removes the synchronism and the redundancy that has the signal for a security transmission. And finally get de coded position. Then it passes this information to the superior layer. With this information the session layer, elaborates the next set of instructions that has to make the next player. And finally show it in the Game board or in another screen.

This diagram is a summary of that

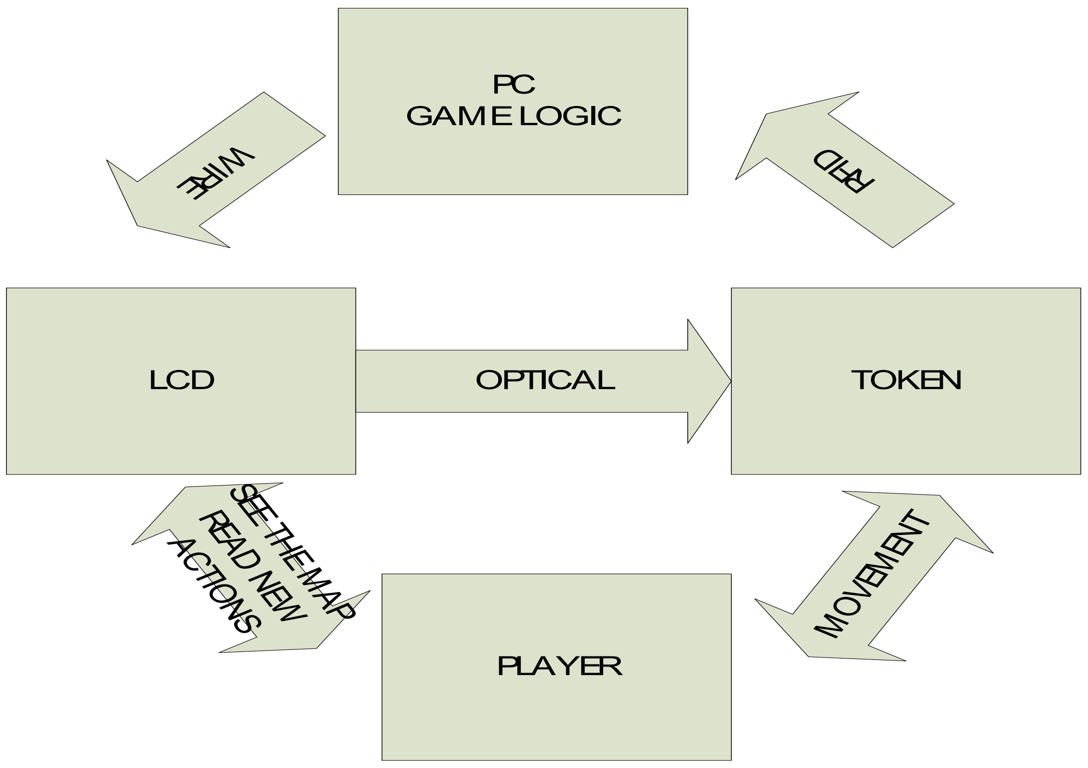

Fig. 2.5 TUI diagram

To sum up, the topics that cover this thesis are painted in green in figure 2.4, we have to find a good codification for the position, this means that we are going to use an smart data structure that using the same bits as other data structures; with this we can code more positions. And then transmit this signal without the player can detect them; finally get the signal and say: “your token is in the door of the castle”. Or something similar but corresponding to our game. This part is the project collect localization in the Tuirender project. A little summary of the others parts follows.

2.3 Game design

The chosen game for this is Roborally, this game is build in Java and a full description of rules, game development programming and manual for players is in Game design : The RoboRally Board Game [3] .

Fig. 2.6 Roborally logo

For us the only thing that we are going to see is the embedded signal that we will put in each game box, the game board is an static image but with some animations, for us this animations will be noise for the signal, but really it's not annoying so much. Because it looks like a very sequential game, so at the end we look for the positions without any animation.

In this game, the robots must find some items placed on the board, the robot's movements are assigned to each robot with cards. The next pictures are some screen captures of this game.

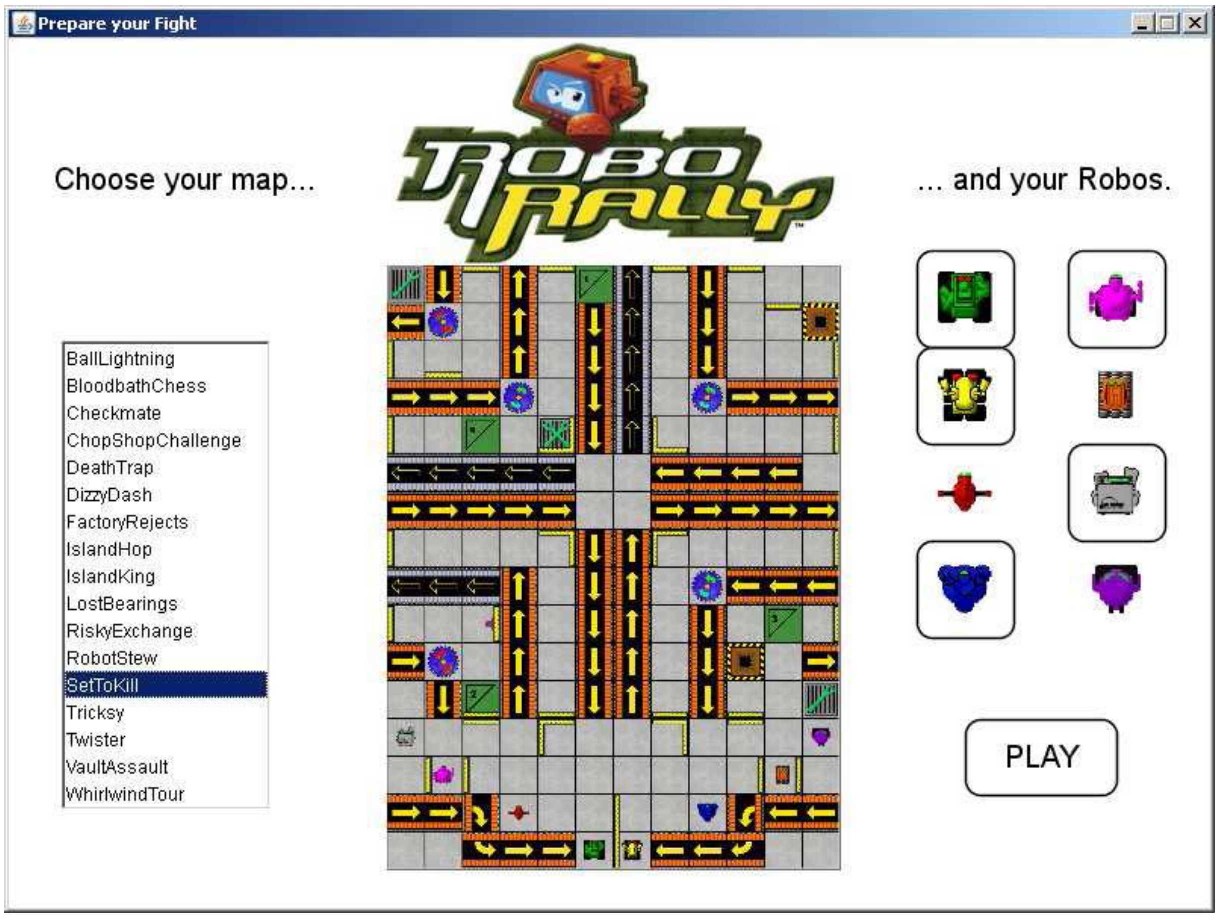

Fig. 2.7 RoboRally beginning

In the above picture and in the next, we can see the beginning screen for the RoboRally game and the game board. For the integration of the localization topic and the game, I think that the easy way is to put one signal in each game box, and then when the robot-token is placed, it will received the signal with the position, another implementation is building another invisible board with the divisions of the screen and then translate the position founded in the invisible board to the real board, in order to procedure with the game

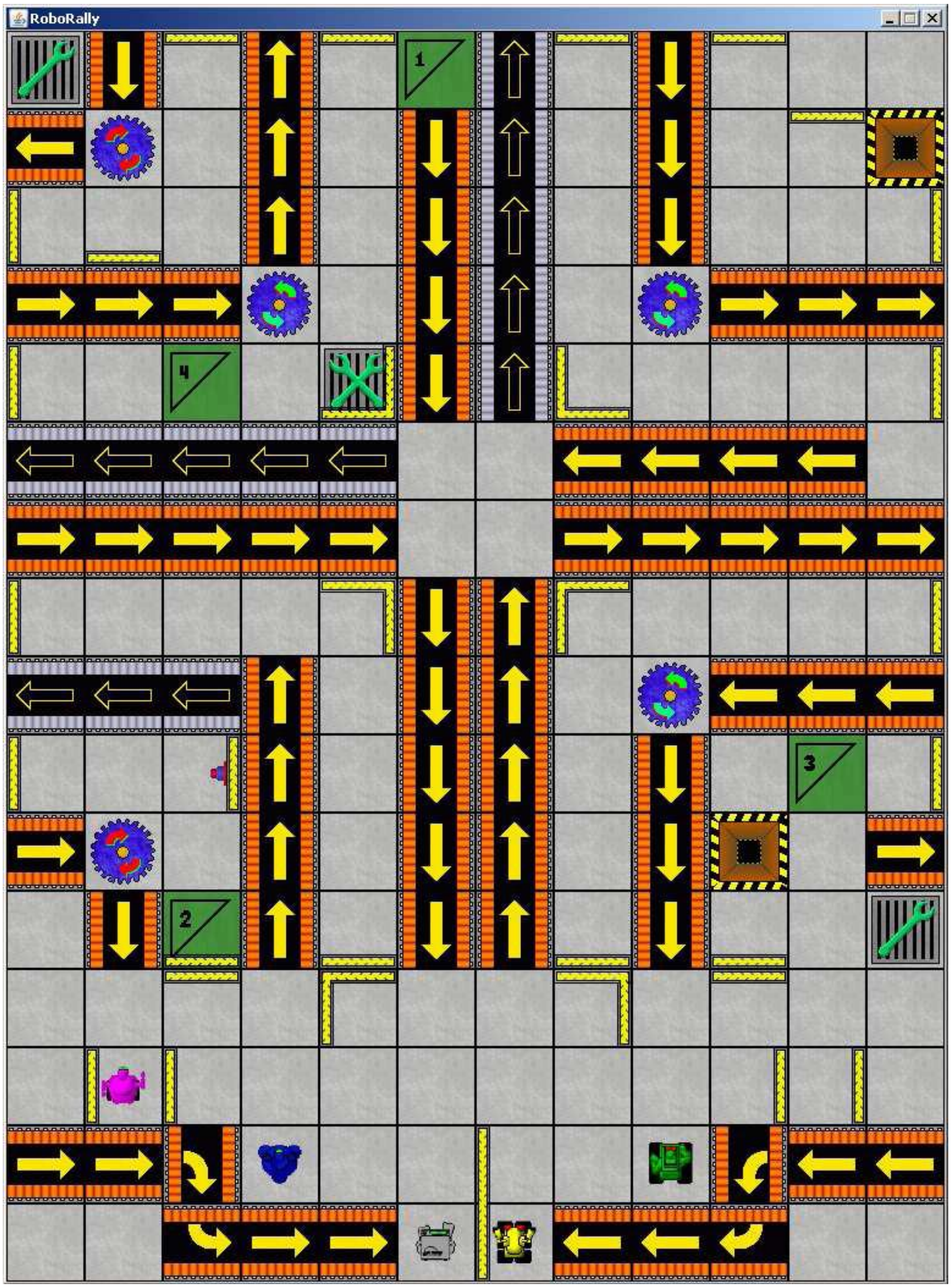

Fig. 2.8 RoboRally board

2.4 Movement

This part, in my opinion, is the most difficult of all the project but it's the amazing one, in this part the tokens has to be able to move around the screen. For make this, the idea is to generate electromagnetic waves that impulse the token in the screen, and produce real movement. Another idea is to use piezoelectric devices around the game board that make like an earthquake in the surface that is placed the token. And these waves carry the token to another site.

This part in this moment is only theory but the experiments are now beginning.

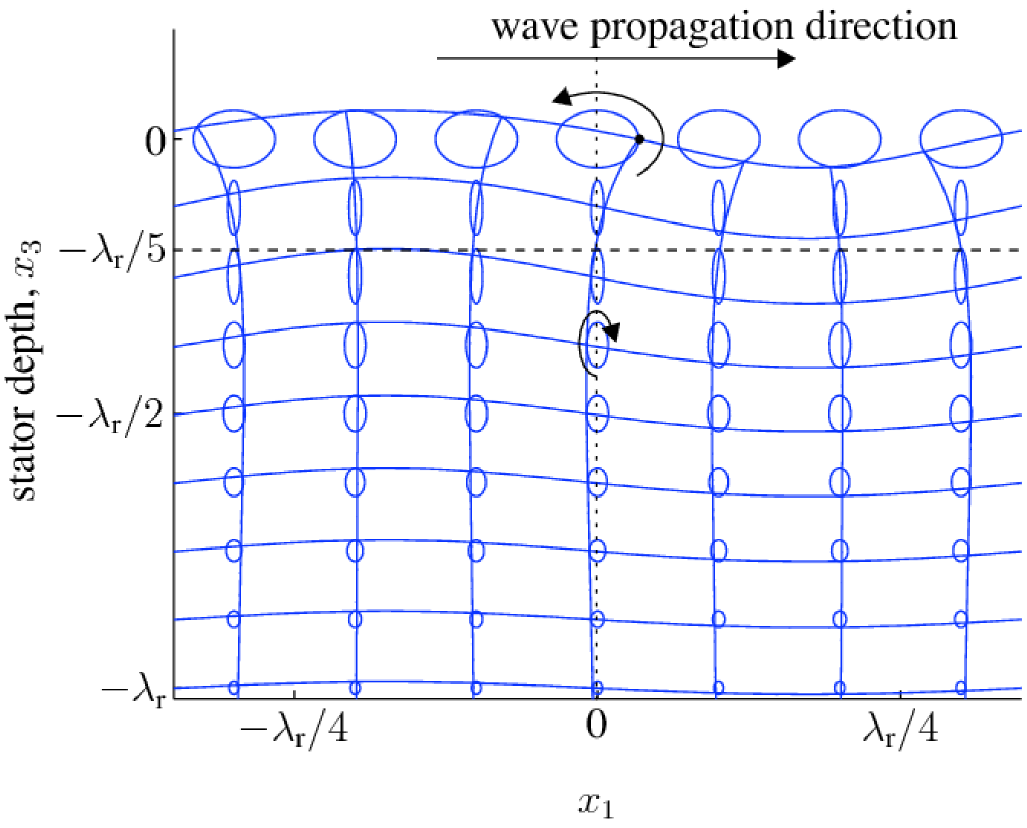

Fig. 2.9 Rayleigh depth

The above picture shows how to create this movement in the surface, by adding little movements on the surfaces that are in the lower levels of the surface that we want to create the movement.

2.5 Token construction

Finally the token construction, it's the part of this project that has made more development inside. In [1], you can find the full description. I also work with this part to really know the characteristics of the signal that I will receive, i.e. sampling frequency, bits per sample, bandwidth to avoid aliasing, and I have to impose some parameters to this part like the minimum sampling rate ..., The last experiments is to work only with a signal that gets the sensor-ADC pack, instead of working with the sensor connected directly to computer's audio device. I show one picture of the token, I hope that you can see the parts that I need to work the sensor and the transinpedance amplifier.

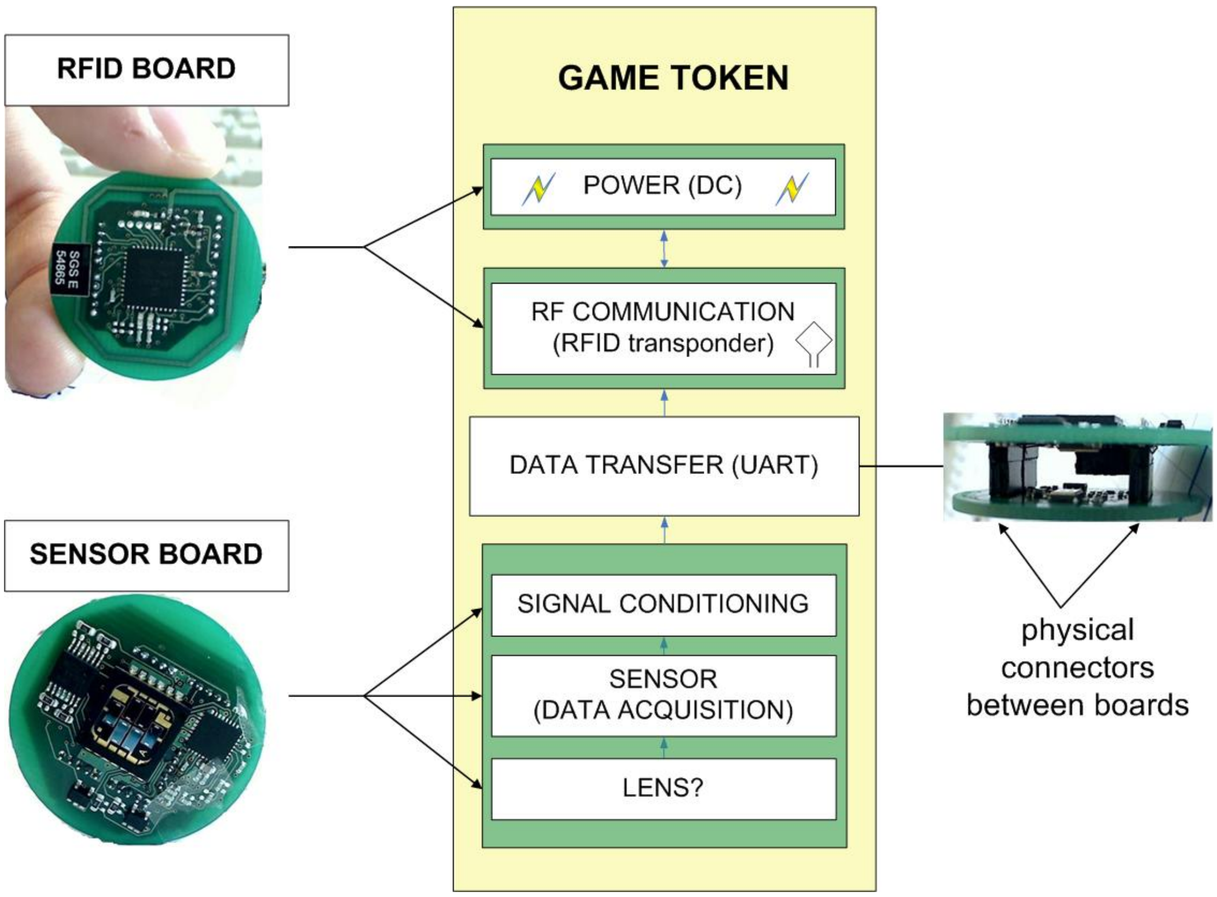
RFID BOARD
SENSOR BOARD
GAME TOKEN
POWER (DC)
RF COMMUNICATION (RFID transponder)
DATA TRANSFER (UART)
SIGNAL CONDITIONING
SENSOR (DATA ACQUISITION)
LENS?
physical connectors between boards

Fig. 2.10 token sensor

Chapter 3

Used materials, inventory

In this chapter I'm going to write a brief description of the items that I used for developing.

3.1 Java

The modulator, which controls the LCD, is all-built in Java, so we use the open source program Net Beans for the Java development, to be honest, I did not make this election but this is the best for making our program multiplatform. The problem that it has is that it works with image and sometimes my machine working with Matlab and Net beans together was so often out of virtual memory. I think that the image processing is not complex, but it wastes a lot of CPU resources sending the new image every screen refreshment. I'm not able to charge images with big resolution and long sequences of data (bigger than 30 symbols). For the future maybe this implementation will be better in a lower level of program language.

3.2 The sensor

I will work with the OPR2100 sensor, this is a six-Element SMD Photodiode array but in my protoboard, I can only use one of this sensors, and this output goes to an AO that amplifies this signal. This schematic has some open issues: We need a symmetric power supply; the operational has saturation output if the input is bright light. And finally find the correct gain for the desired dynamic range.

I believe that another issue is to place here a low pass filter, with the cut frequency of one half of the sampling frequency. This is for the correct sampling of the signal, and to avoid problems with the aliasing that we can have with the signal, if we do not respect the Nyquist theorem, which says that the minimum frequency for sampling one signal and after recover eat is the double of the band width of this signal.

To end with this, I show here the proto-board and the schematic for this circuit

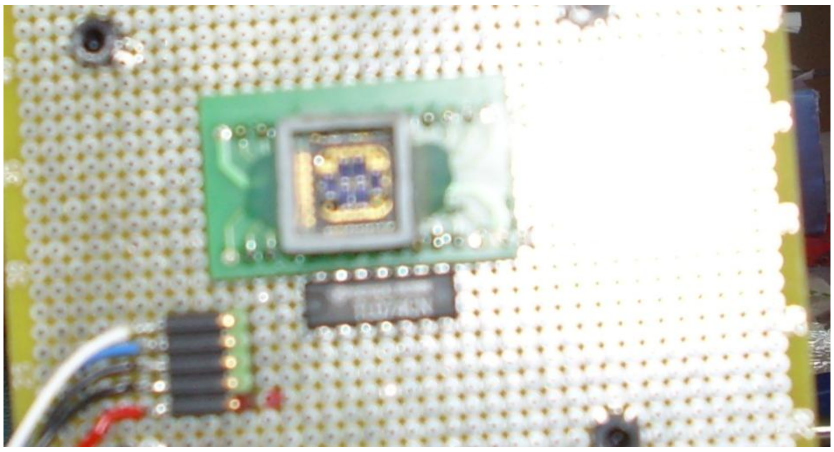

Fig. 3.1 proto-board

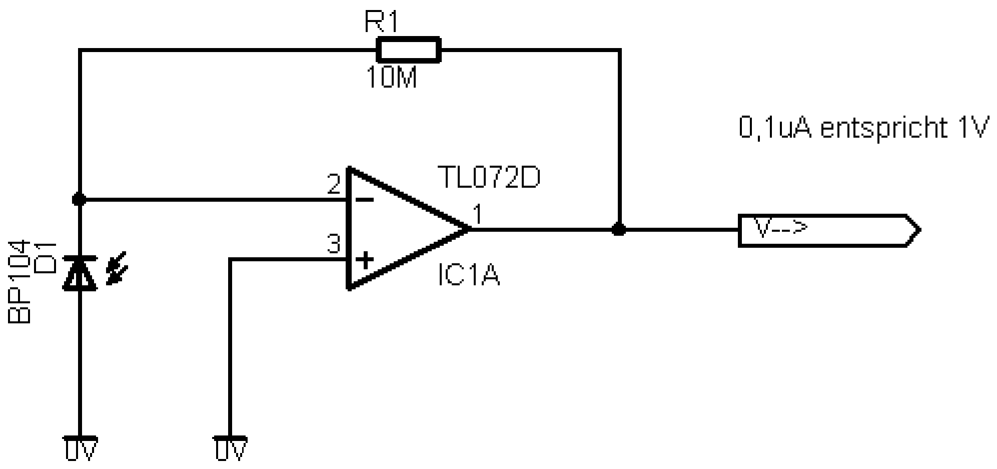

Fig. 3.2 schematic

For the sensor, I want say that the response is good for the visual spectrum, a little focus on the red part of the visible light spectrum, however this will not be a problem, because thanks to that we can put our signal only using the red component in RGB system, and this will be more invisible for the human eye.

Here is the response of the sensor

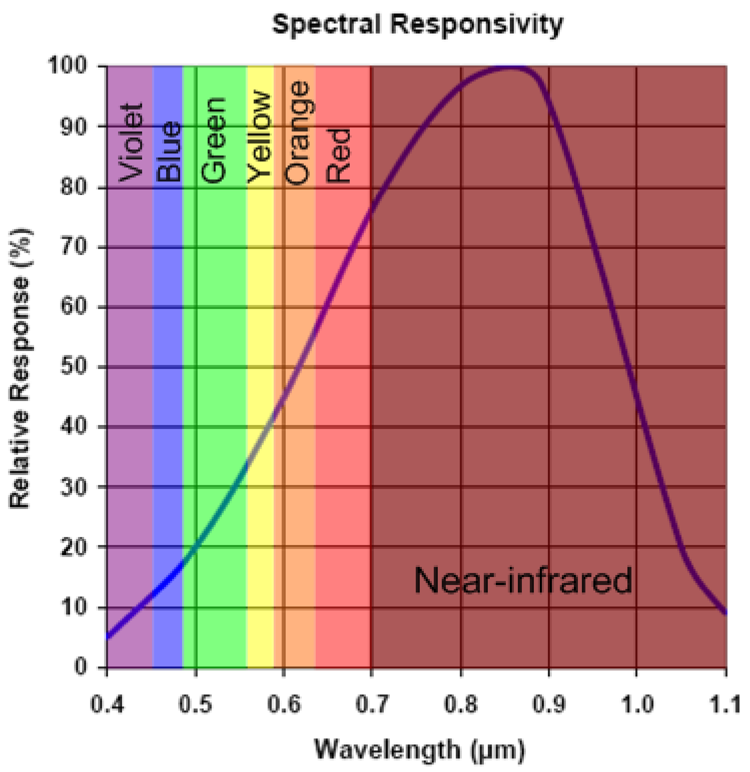

Fig. 3.3 sensor spectral responsivity

3.3 The LCD

The LDC display that we are going to use is the Samsung Sync Master 193T, this is a normal screen for a PC, this was chosen because if set full brightness we do not have Backlight noise which is a good advantage, furthermore in this thesis we also can demodulate the signal with some of backlight noise, we can add it if we put the screen bright not in the maximum position.

The maximum refresh rate that we control is 60 Hz, It's very slow so for new applications will be good to use another screen with faster refreshment rate.

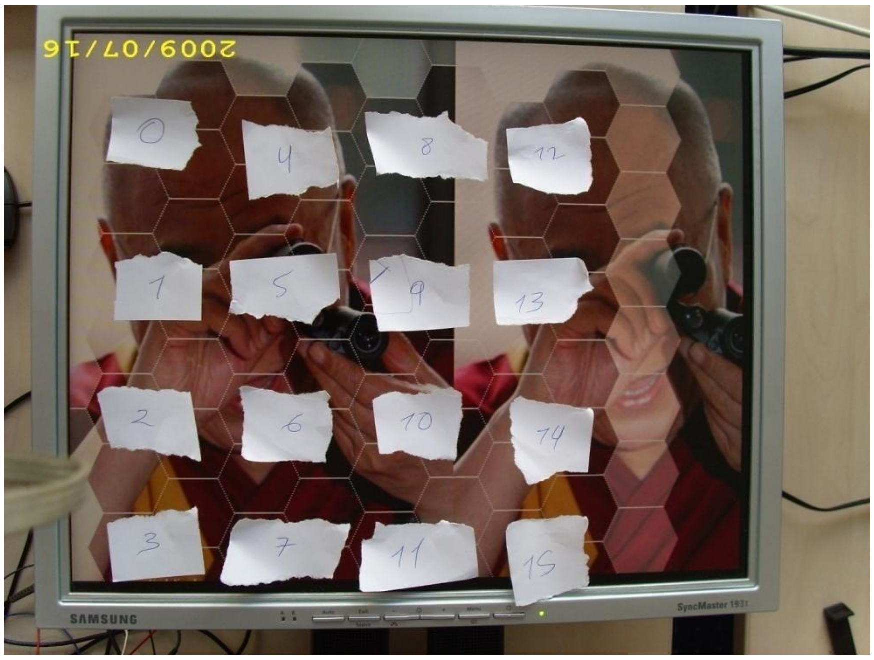

Fig. 3.4 LCD used

The spectrum of the signals that send the LCD of Samsung display family is

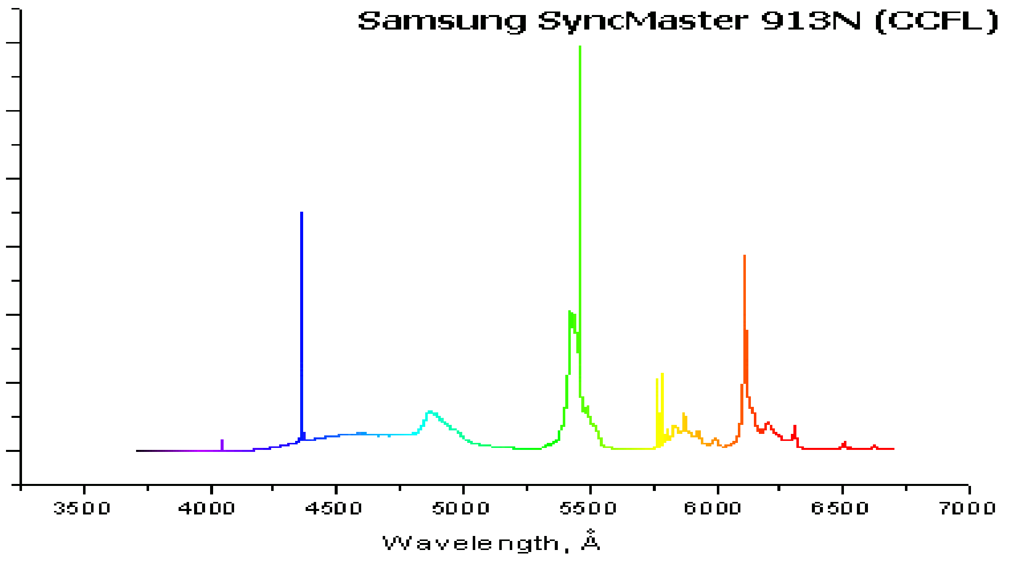

Fig. 3.5 response of LCD

From one simple inspection of this picture we can deduce that the sent signal and the receptor work in the same ratio of frequencies.

Finally, the unique thing for making this thesis must know is well explained in the Wikipedia site [6], really not more data is needed. That the reason that I do not explain, how an LCD works, because there are a lot of explanations in the literature and I did not make a very intrusive study of this topic.

3.4 Audio device and Matlab

The audio device that I use is the USB AUDIOCAPTURE UA-25 EX, it can provide sampling frequencies until 96KHz and precision of 24 bits, the chosen sampling frequency is 48Khz and we can work with a down sampled signal, If we put the audio device at other frequencies we do not have good results, so we get this signal and then make downsampling using a good software anti-aliasing filter which I think that is the main problem here.

We have to forms on managing this device with the Java project Masound and from Matlab commands, the second is a non finished project so we can not set any parameter only get sampled signal at 48 KHz, and the other is directly from Matlab commands, both of them are good for this project.

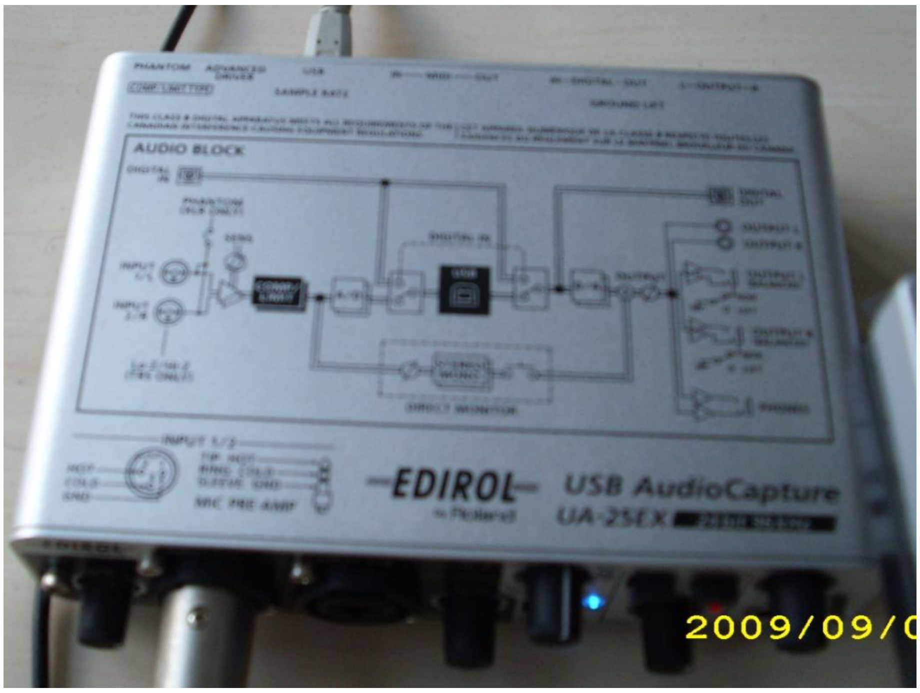

Fig. 3.6 Audio device

3.5 Laboratory

For the power supplying I use the DC power supply PE1542, from Phillips, This is used, basically, to get de DC power that needs the OA

And in order to get fast measurements of the signals we also use the four channel digitizing oscilloscope TDS 510 A from Tektronix.

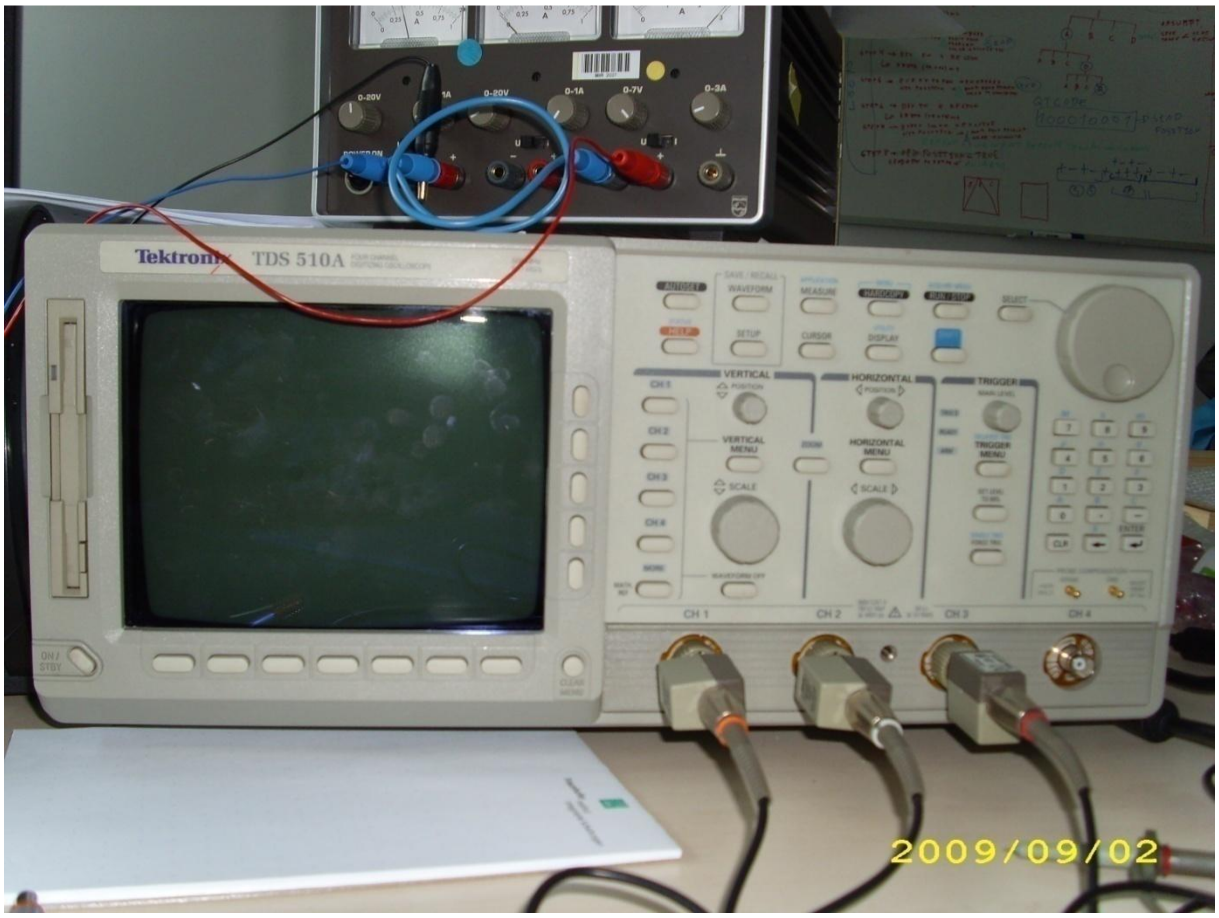

Fig. 3.7 lab materials

Chapter 4

First experiments

In this chapter I'm going to show you some experiments the most important ones, and measures that I made, first of all I want to say that this is not an extensive study of the LCD displays, there are only some experiments in order to characterize the signal that comes from the LCD to the token.

For the response of the sensor you can look at [1], and there you can find an extensive analysis for features like difference sensitivity for each colour …

4.1 System Identification

In order to find the response for our system, I will use two things; one is send a delta signal, for this I will send first 30 black frames then one white frame and finally 30 more black frames.

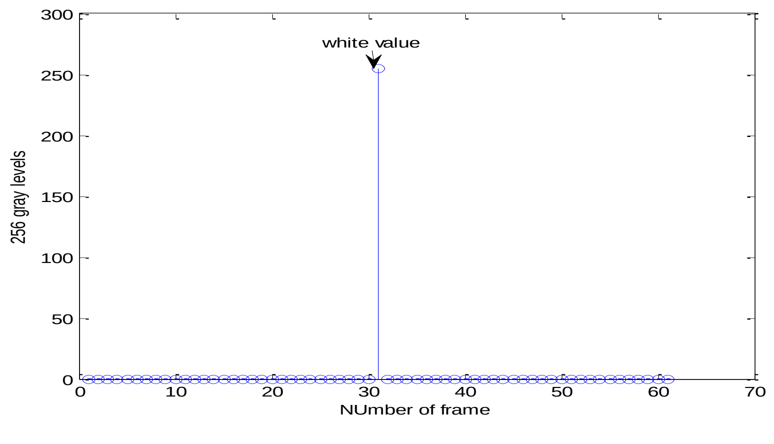

Fig. 4.1 sent delta

And then I will catch the received signal. This signal is one second long and has 60 frames 1 white and the other black, now you can see the received delta. Then we can say that the signal is causal because is zero but it's not zero after the delta, let's find its response in frequency.

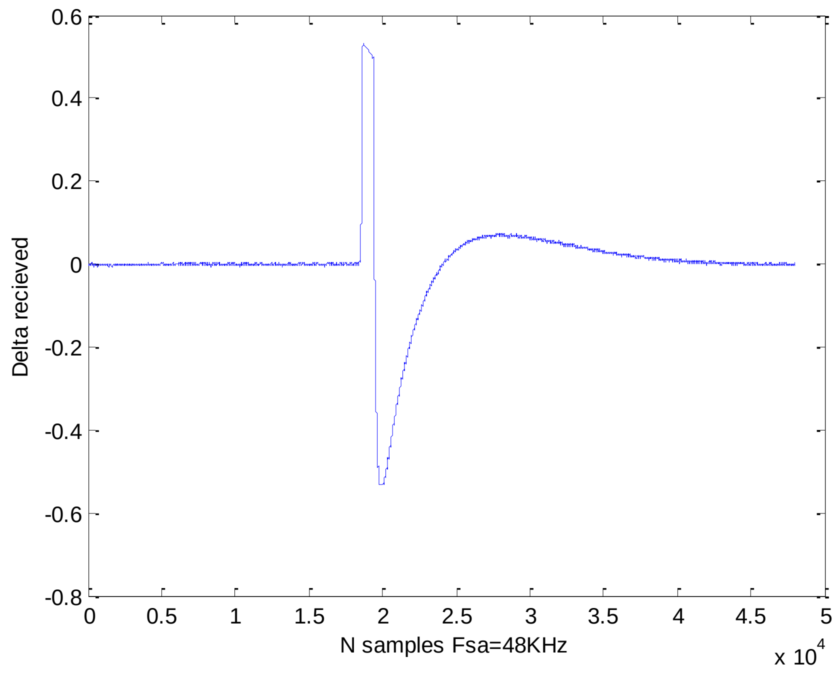

Fig. 4.2 Received delta

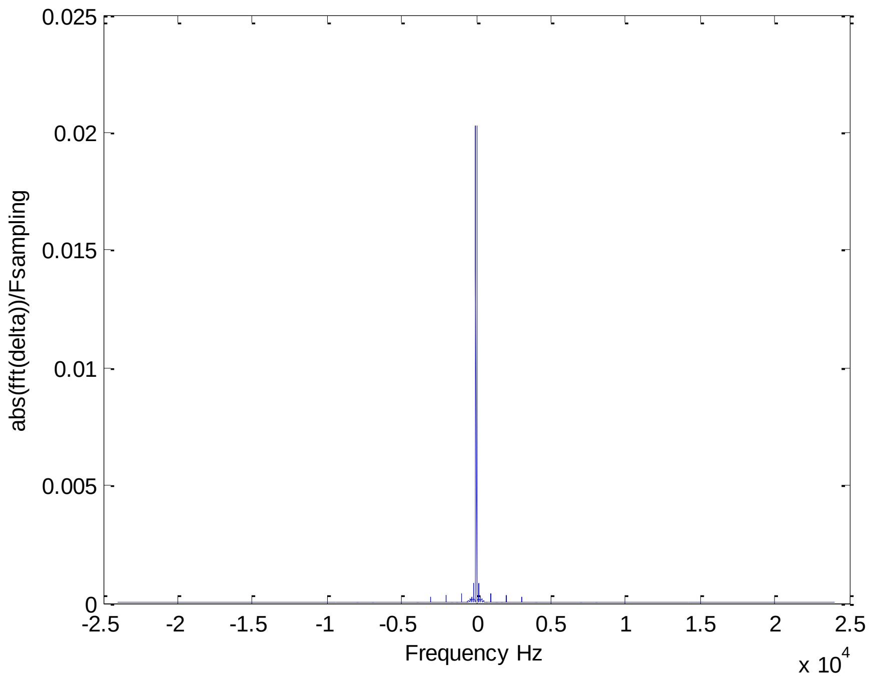

Fig. 4.3 FFT

In the last picture we can see that our system has response in the low frequencies and it's a narrow band system.

Now we are going to see the FFT but with making the logarithm and some zoom.

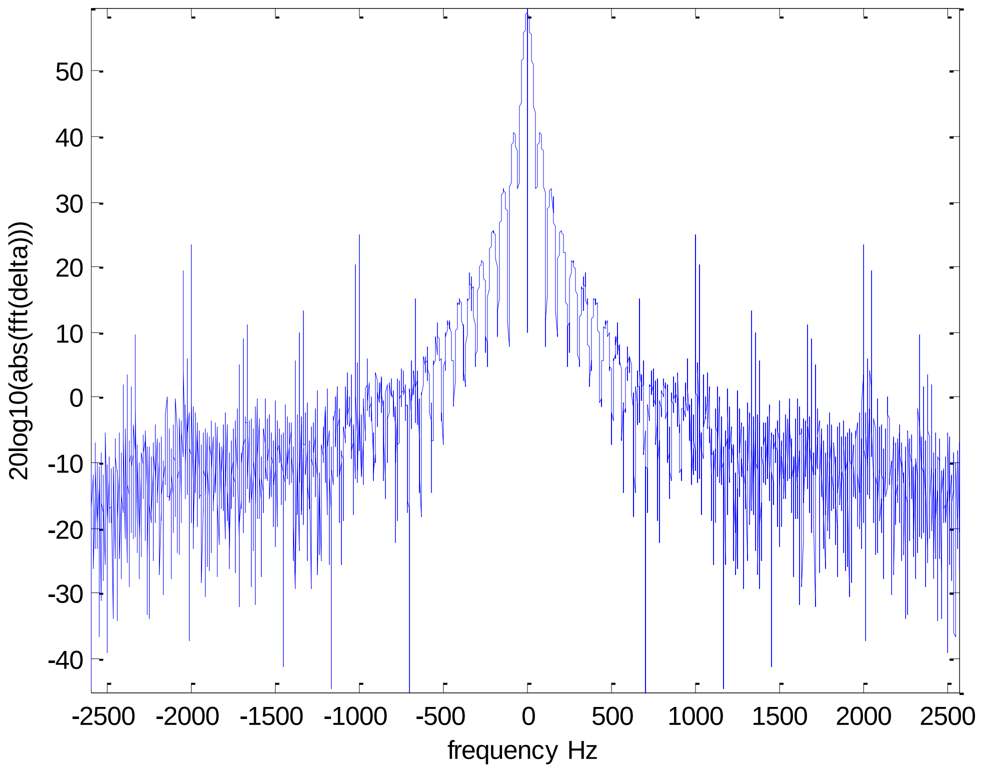

Fig. 4.4 FFT

Now we can see that has lobules, it is decreasing except our in the middle the low frequencies, finally I'm going to make more zoom in order to define some parameters.

In the next figure we can see that the lobules of that sinc is near 60 Hz, this is in that way due to the refresh frequency of the LCD is at 60 Hz, so I 'm getting here the frequency response of one signal that is a digital pulse this is the reason of this spectrum looks like a sinc. So the duration of our pulse is 1/60=16.7 ms. Finally you can see that there is a pole in the zero frequency, this pole is caused by the audio device, but will be better for us not having it, because our signal will be in the range of 0 to 60 Hz. Finally I repeat the experiment but adding the back light noise to the signal. The back light is produced because the LCD cannot turn on at the same time the LEDS from red, green and blue, it generates some high frequency when it turn on and turn off this LEDS during the time that it shows the picture on the screen.

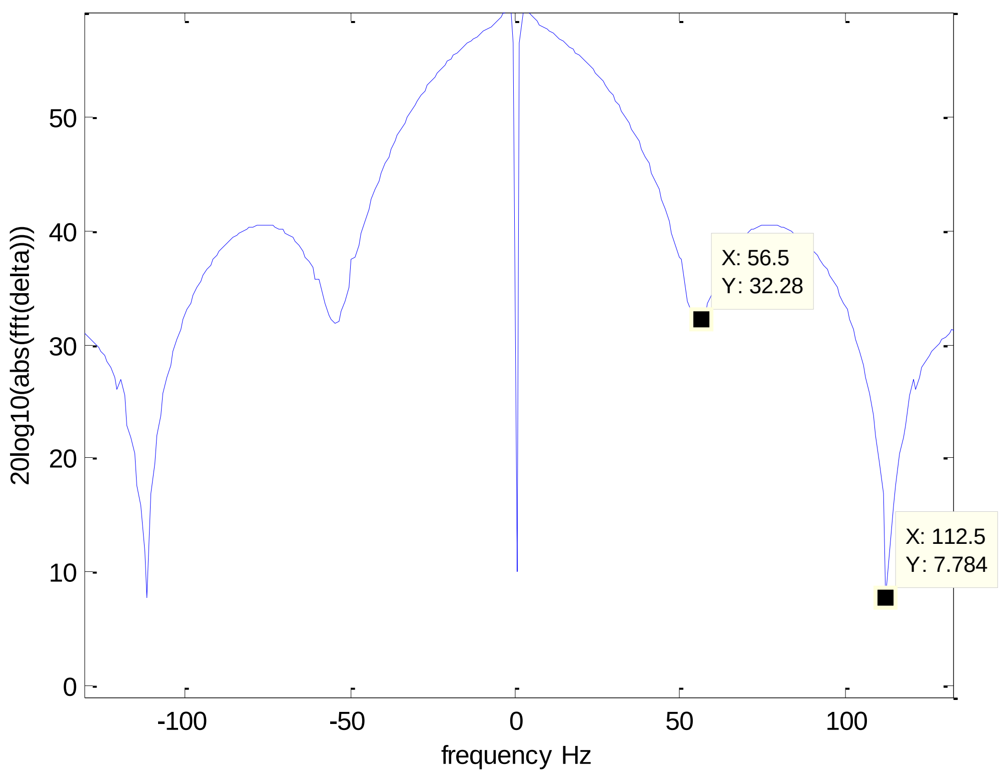

Fig. 4.5 Zoom in FFT

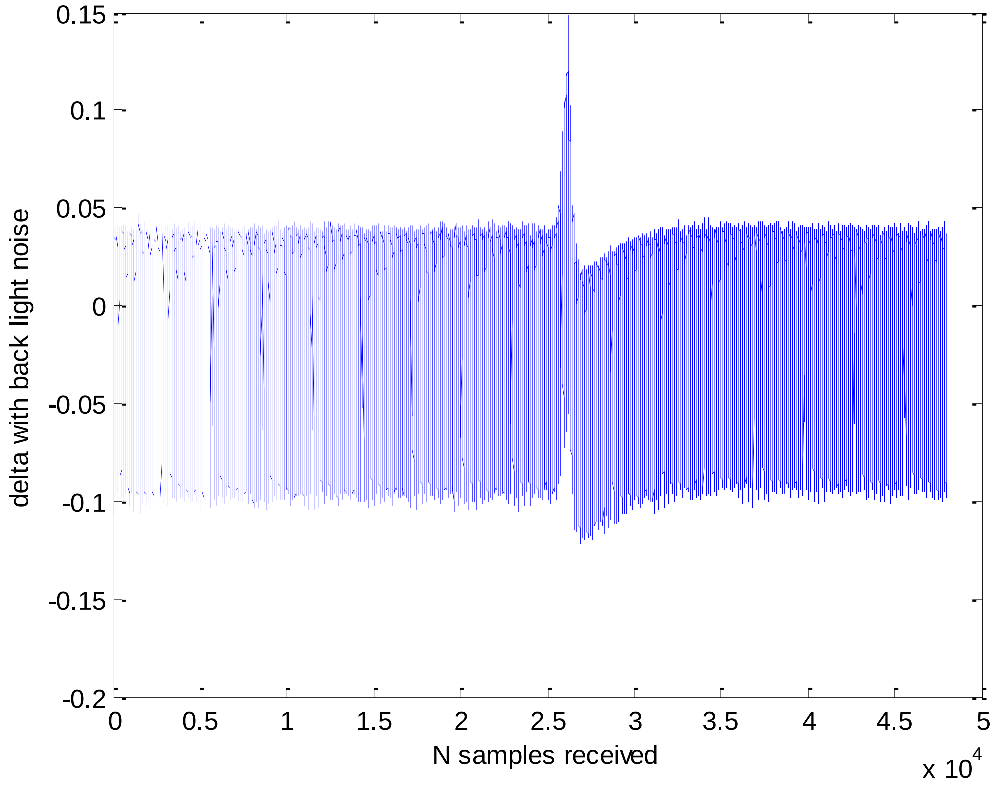

Fig. 4.6 delta with back light noise

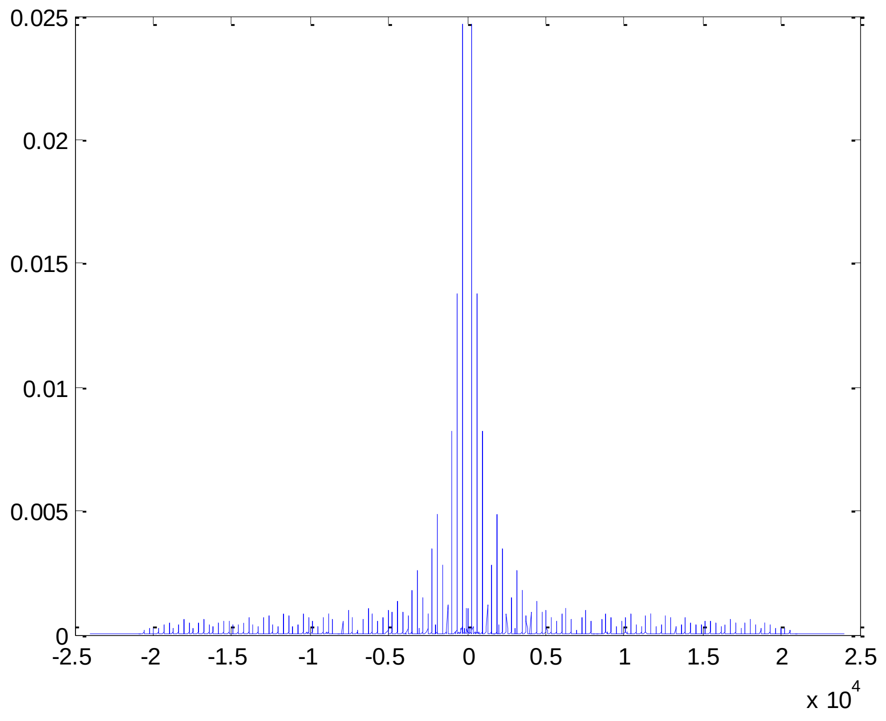

Fig. 4.7 FFT of delta with backlight noise

In the above picture, you can see that this signal has more harmonics, that without the backlight noise

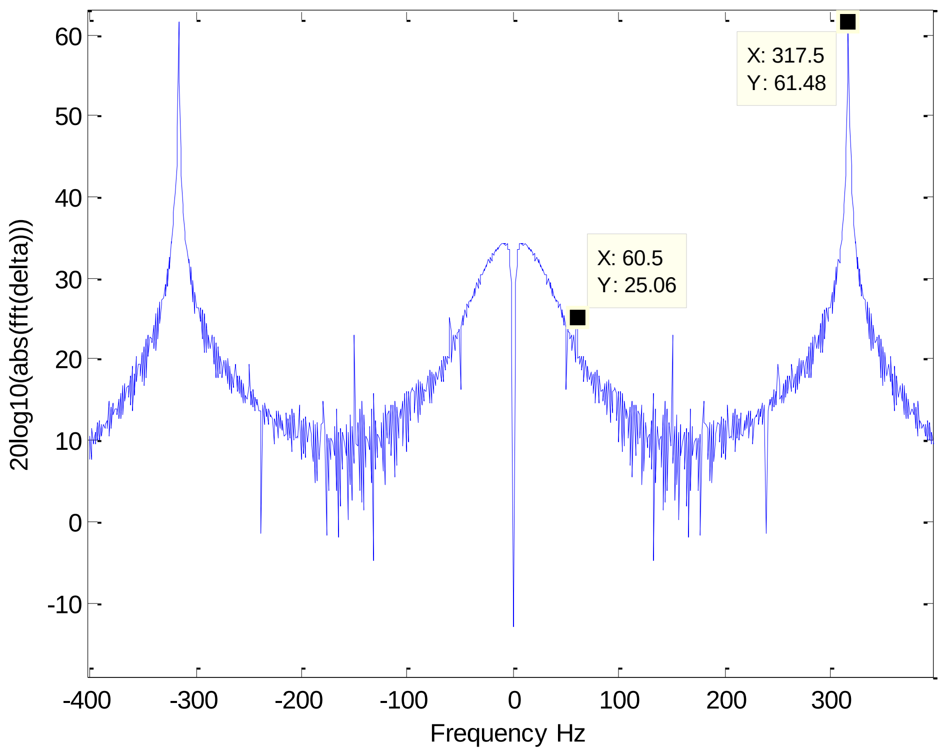

Fig. 4.8 logarithm of the FFT

In the last picture, you can see what exactly produce the backlight noise, basically a tone centered in 317 Hz, so you can deduce that all the information in the range of 0 to 60 Hz, so an idea for remove this it's use a low pass filter.

In other to see better the FFT of the delta, I'm going to winnowing this received signal with the hamming window.

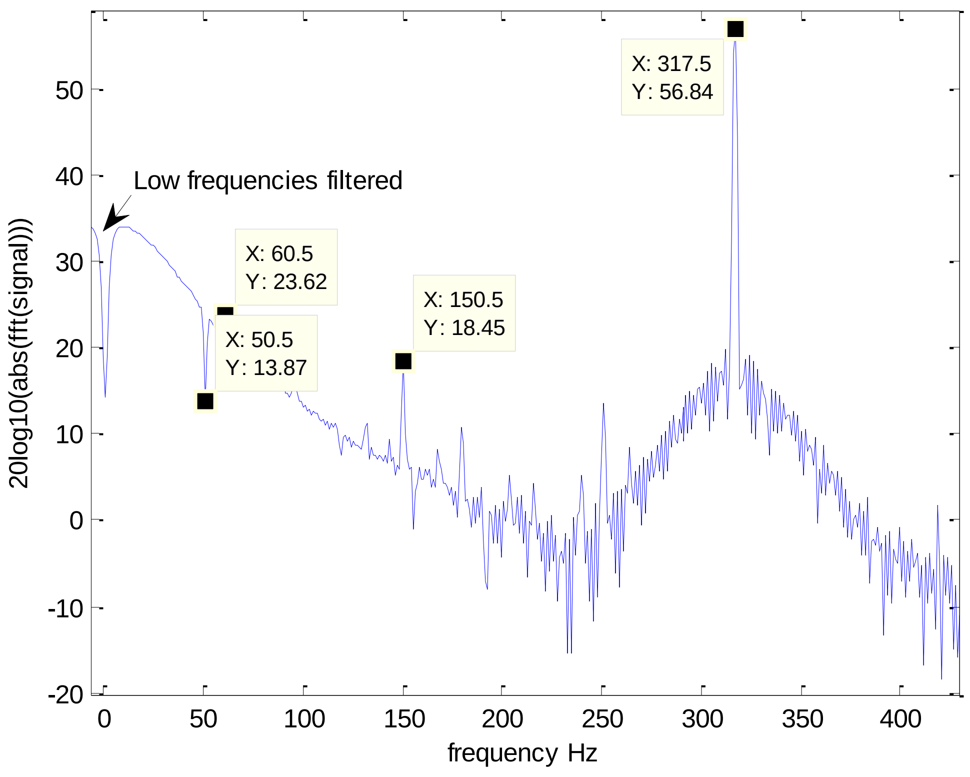

Fig. 4.9 logarithm of the FFT

In the above figure we can confirm our expectations, the peaks are narrower and centered to this frequencies, and you can see that there are filtered values near 0 and 50 Hz this is due to the audio device. But you can see that the information between 0 and 60 Hz is easy recovered using one low pass filter to filter the peak at 317 Hz

Another experiment is to send random values to the display, in this case I will send random gray levels, this gray are in the rage of 120 to 150, this range is inside the gray range of 8 bits gray, which means 0 is black and white is 255. If I send this the signal that I receive is like that

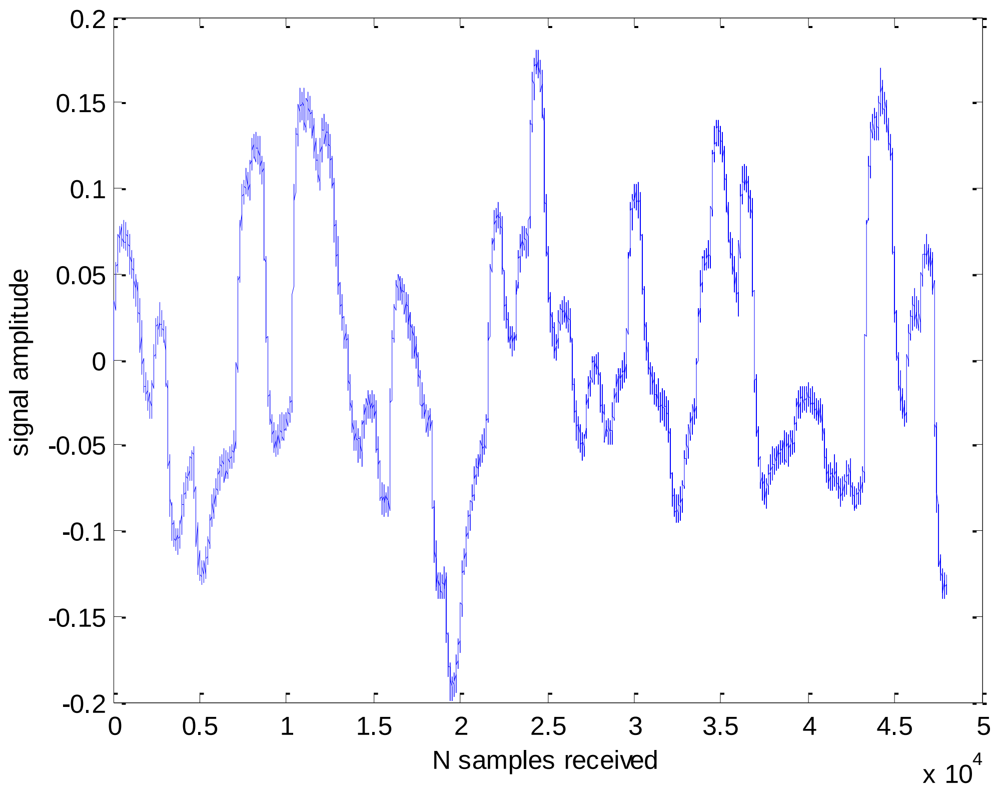

Fig. 4.10 Random gray levels.

Looking at the above signal, we can easily consider that this is a base band signal, some kind of PAM.

To make some analysis, of this random signal, first I consider the autocorrelation function. I hope that will be similar to a delta function, because I put some random signal at the input of the system.

If you look the next picture you can see, that this function is not a delta but is quite similar, this is because as I said before this system makes a filter in low frequencies, and really our signal is random but limited to 60Hz.

To end with this analysis, I need to make the spectral analysis of this signal, like the signal in the input is random; I'm going to make the periodogram of this signal.

As I expected, in figure 4.11 you can see that the signal is low pass and has one peak near 60 HZ the frequency of refresh, and this is also the cut frequency because the signal has fall 3 DB from the maximum point. So the system is low pass with the maximum frequency of 60 Hz, this frequency is always the refresh frequency of our screen. Finally I will show the same periodogram, however this time I will add the back light noise. This time the band with that is necessary for me is from o to 350 Hz. Because I expect a tone in this frequency (317 Hz)

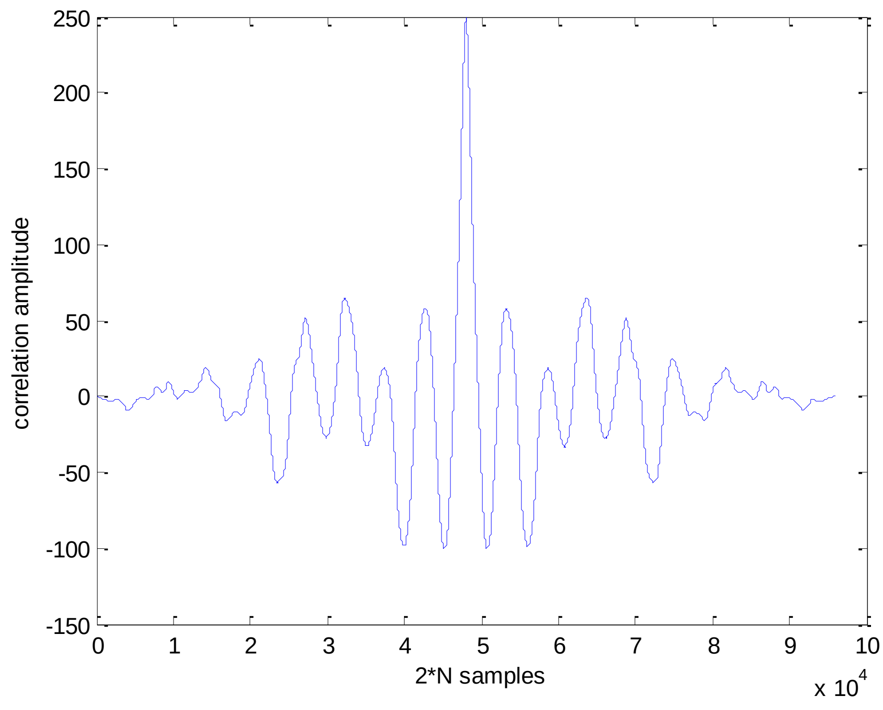

Fig. 4.11 correlation of random gray levels

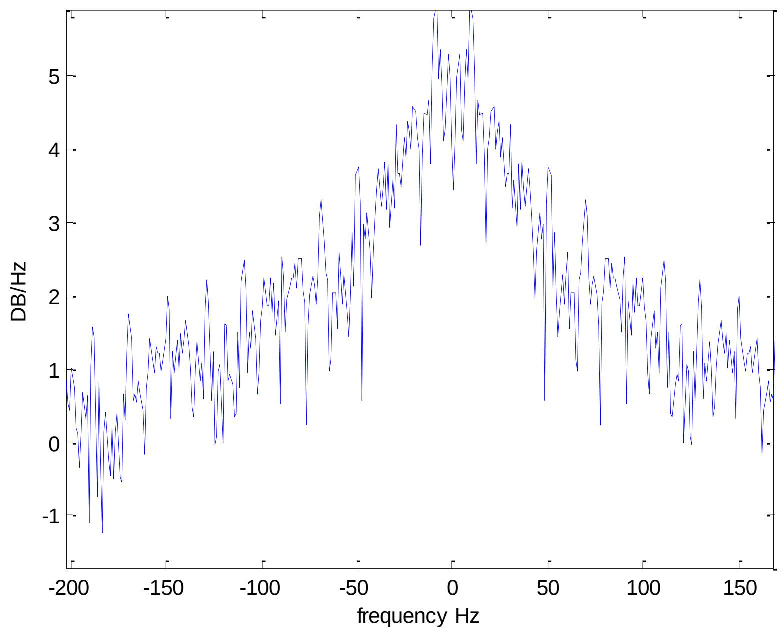

Fig. 4.12 Periodogram of random greys

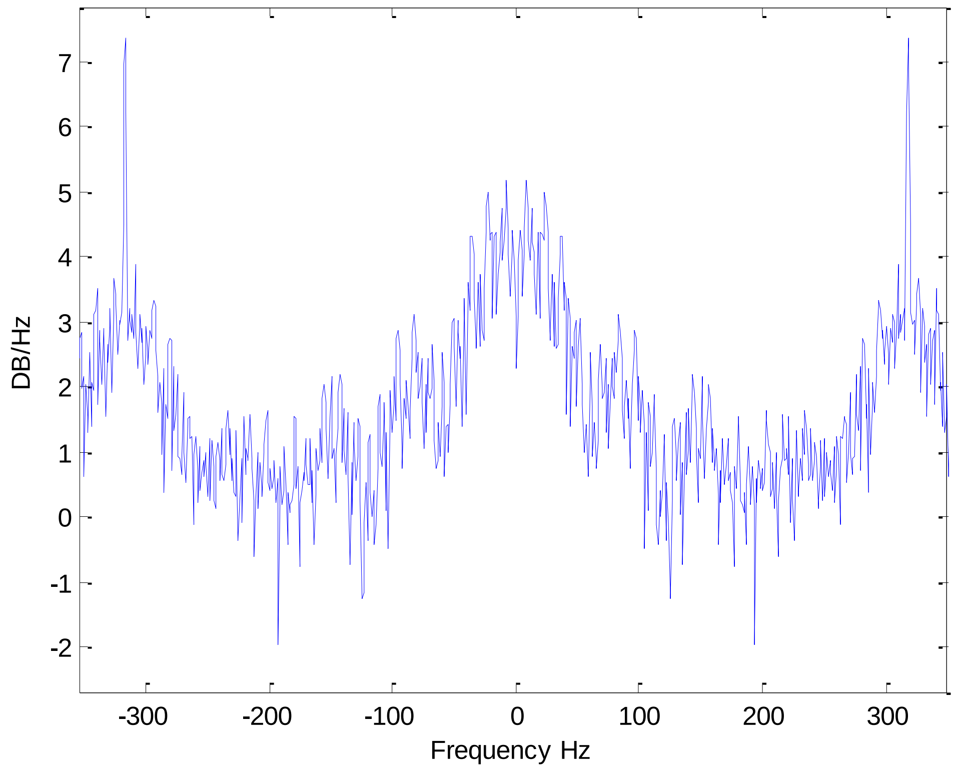

Fig. 4.13 Periodogram of random gray levels with back light noise.

In the above figure, we can see the peaks produced by the back light noise, near 320 Hz, but the important thing is that our channel in 0 to 60 HZ is equal to the channel without backlight noise, so we can filter that, in order to remove the back light noise.

To sum up, I find that the system is similar to one base band digital system with frequency rate of the digital symbols of 60 Hz, this frequency always is the same of the refresh frequency of the LCD.

4.2 Sending signals to the LCD

In this part of the chapter, I'm going to show some non linearity that has the screen, for this I'm going to send some signals to the screen one will be a binary PSK and then a ternary PAM

In the next figure, you can see that I send one signal that is this sequence repeated 01111, and then I'm going to show the problem. This problem only passes if we have continuous values, at one short description is if the LCD has to show one image with low values and then I change the image to one that is higher, (more brightness), if the next frame has the same level, this firs one always will be lower. Take a look on the captured images. The sequence from the second is -1, 1,0,1,0. For sending this sequence I send a red image of RGB (120,0,0) and the modulated it by my signal so I will always say the red image but with this levels 1 (119,0,0), 2 (121,0,0), 3 (120,0,0), 4 (121,0,0), 5 (120,0,0)

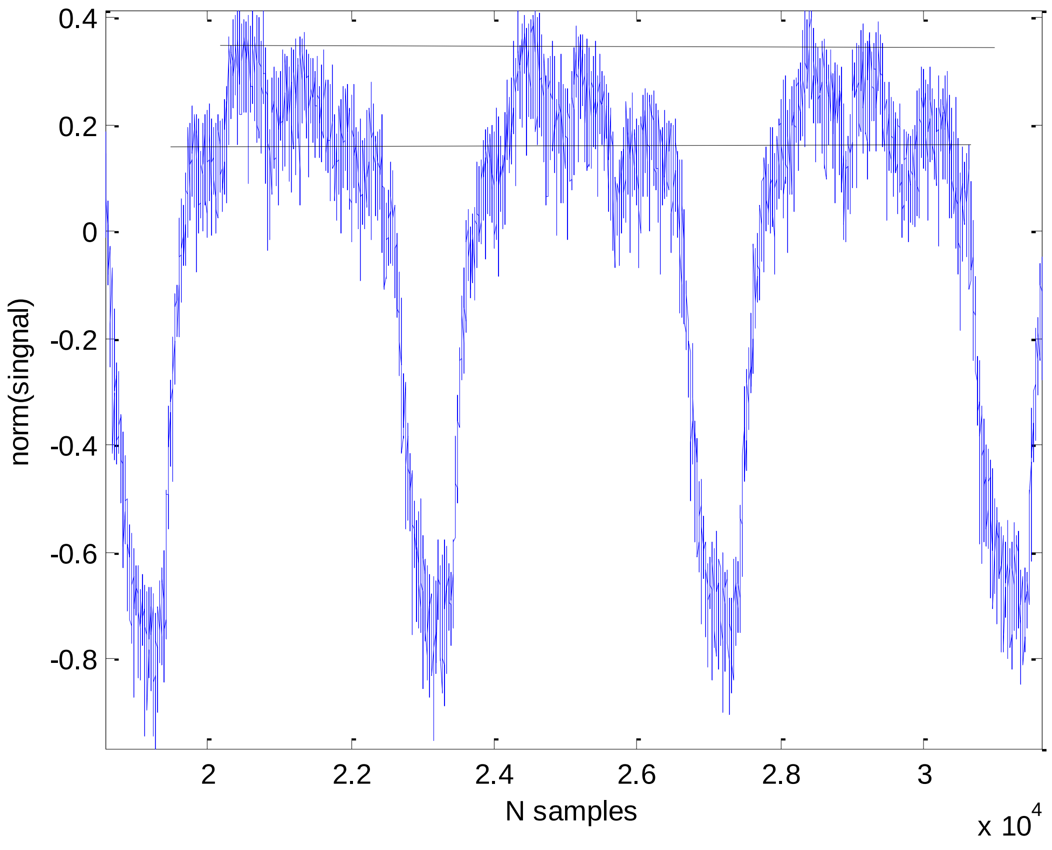

Fig 4.14 sequence of 0, 1,1,1,1

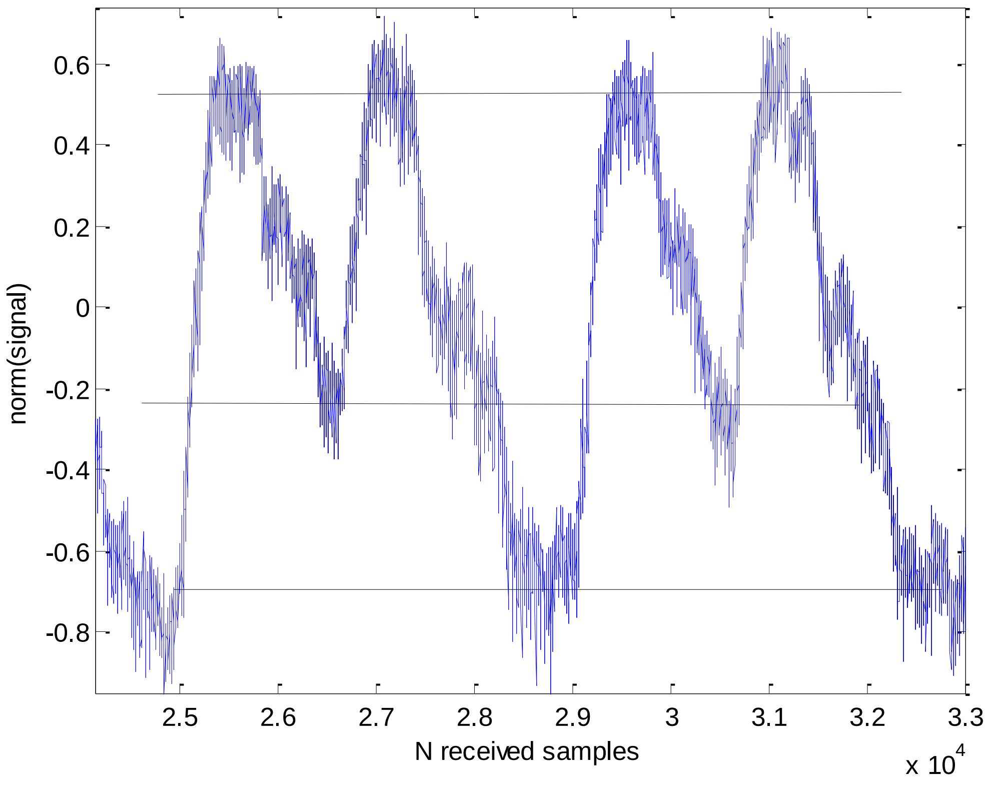

Fig. 4.15 sequence of -1, 1,0,1,0

In the last from the last figures, we can see that if our signal has changes then the amplitude is the same for the same RGB values, but when the signal is quite constant, this is not true, so it's important that our signal has a lot of changes, because if it is quite constant,, we will have the problem that the same value in the RGB system will be displayed as different levels of brightness.

Finally I'm going to show one figure that really shows the problem of the backlight noise

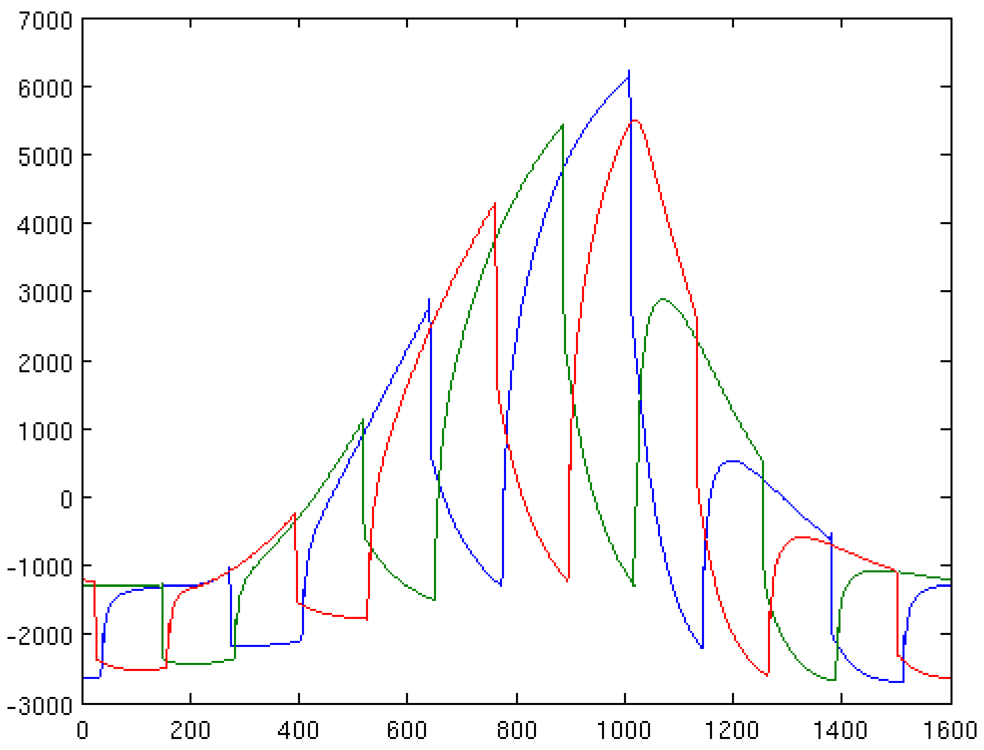

Fig. 4.16 Back light noise

The figure above shows two frames of levels 127 and 0 at 1/60 seconds per frame (800 samples per frame at 48 kHz). The display back light is modulated at 130 Hz (48000/130 = 369.23 samples per period). The waveform plot follows a cyclic pattern of the red, blue, green, red, blue, green, red... lines.

Finally, I'm going to show two sent signals with backlight noise and without this are first filtered and then sampled to the correct sample rate, which is the symbol interval, you can see that really remove the backlight noise and the noise that have the signal. For that I use a Low pass filter of cut frequency of 60 HZ.

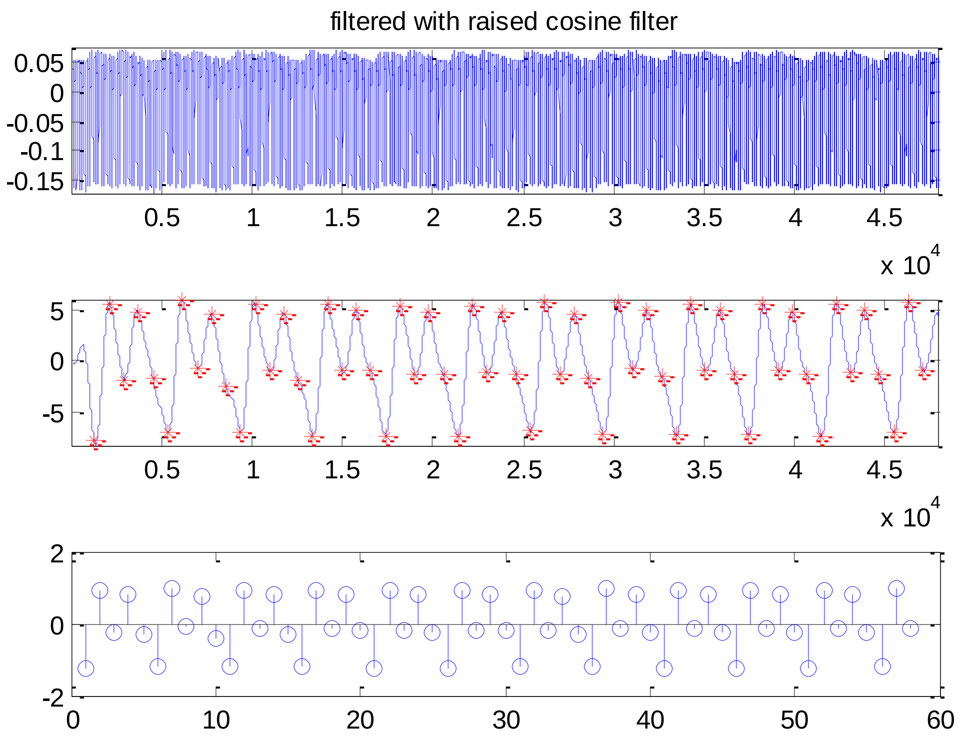

Fig. 4.17 Filtered signal

The figure above shows that we have one signal with Backlight noise, and I apply a low pass filter at 60 Hz, then you can see that, the backlight noise disappears, for better explanations of that take a look on the next figure, I make a zoom in.

In the next figure we can see that the signal has some kind of AM modulation at the carrier frequency of the backlight noise, so when I filter this signal it disappears. Because I can get the information that is carried by the back light noise.

To confirm this take a look on the spectrum of this two signals. The received and the filtered.

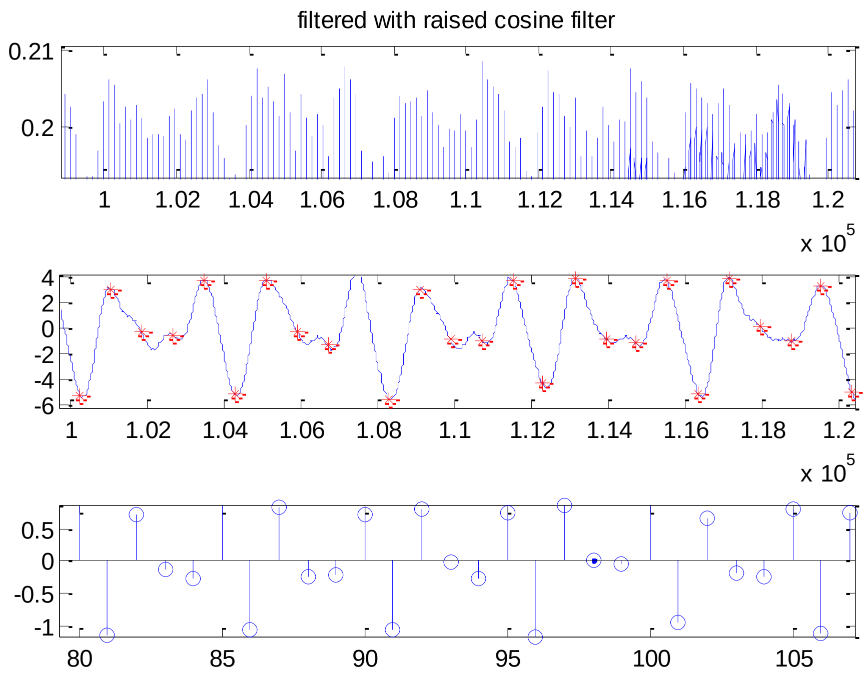

Fig. 4.18 Zoom in filtered signal

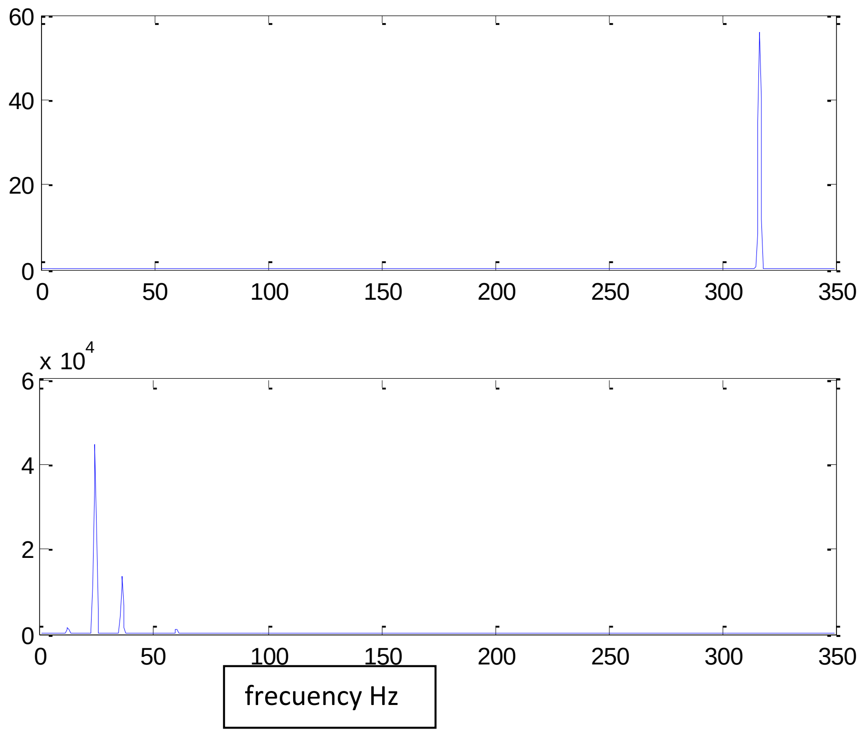

Fig. 4.19 Spectrum of received signal and filtered

Last figure shows the range of frequencies from o to 350, we can see that the first has a tone on 317 Hz, but when I apply a Low pass filter this disappears, and only has the information.

Finally I'm going to make the same but, this time; the signal will not have the backlight noise.

You can see that the recovered signal and spectrum is the same as the signal as I showed before.

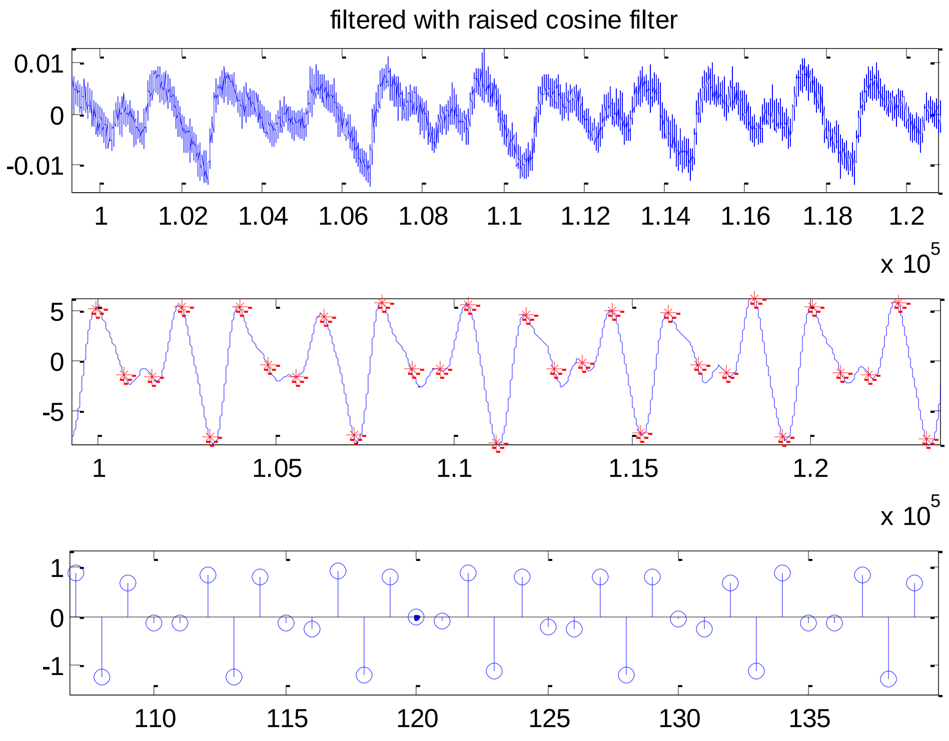

Fig. 4.20 received signal without backlight noise

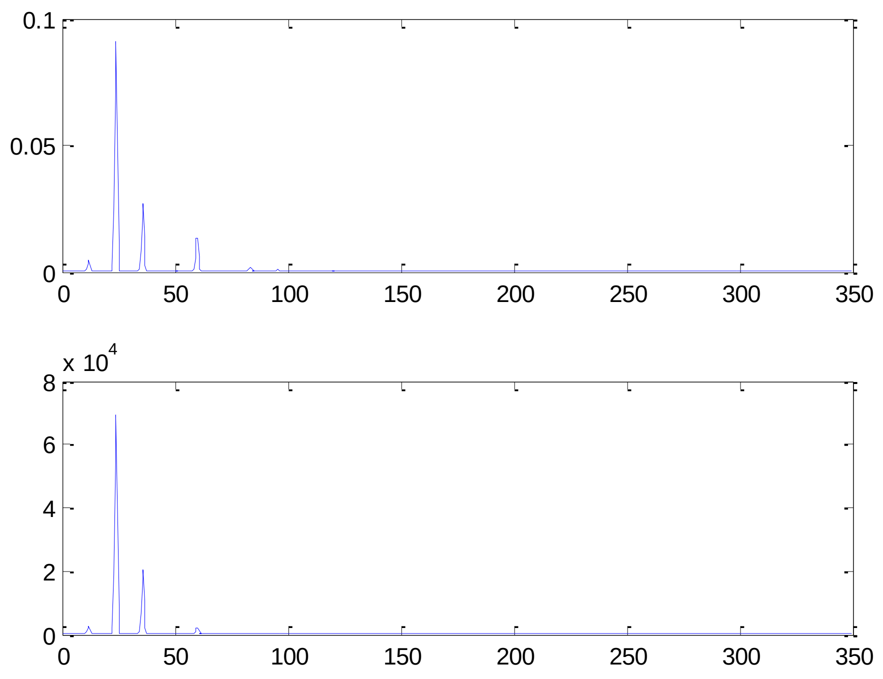

Fig. 4.21 spectrum of received signal without backlight noise

4.3 Noise in a displayed image

Finally, this measure will indicates which are the properties of the noise that the sensor receives while in the screen is displayed an static image. This image is loaded by the java JFRAME.

In order to measure the properties I plot the autocorrelation

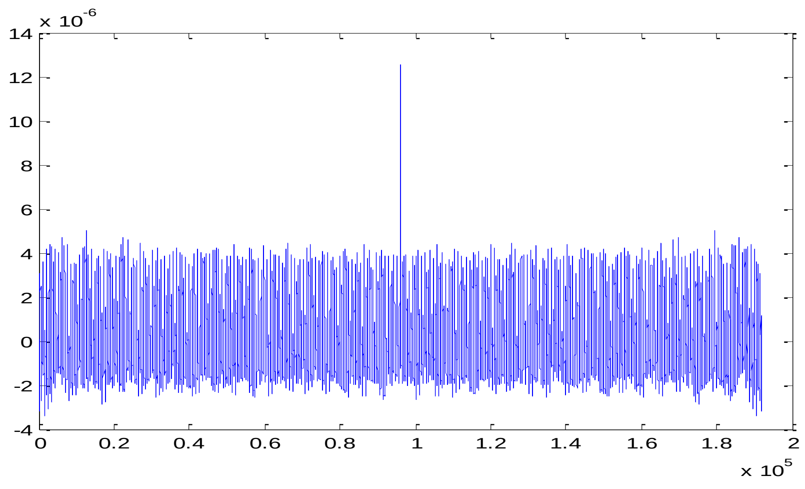

Fig. 4.22 autocorrelation of the noise

In the last figure you can see that is the expected plot for an AWGN so I will model the noise of this type the power is 12e-6 Watts, so really is very low the problem is that when the image changes the screen ads more.

Finally you can see the density spectral power of this random process. The important thing here is that is bigger at the refresh rate, and in 50 Hz, this is coming from the power net. The red mark is at 60 HZ and the green at 120 and finally the yellow at 180 Hz.

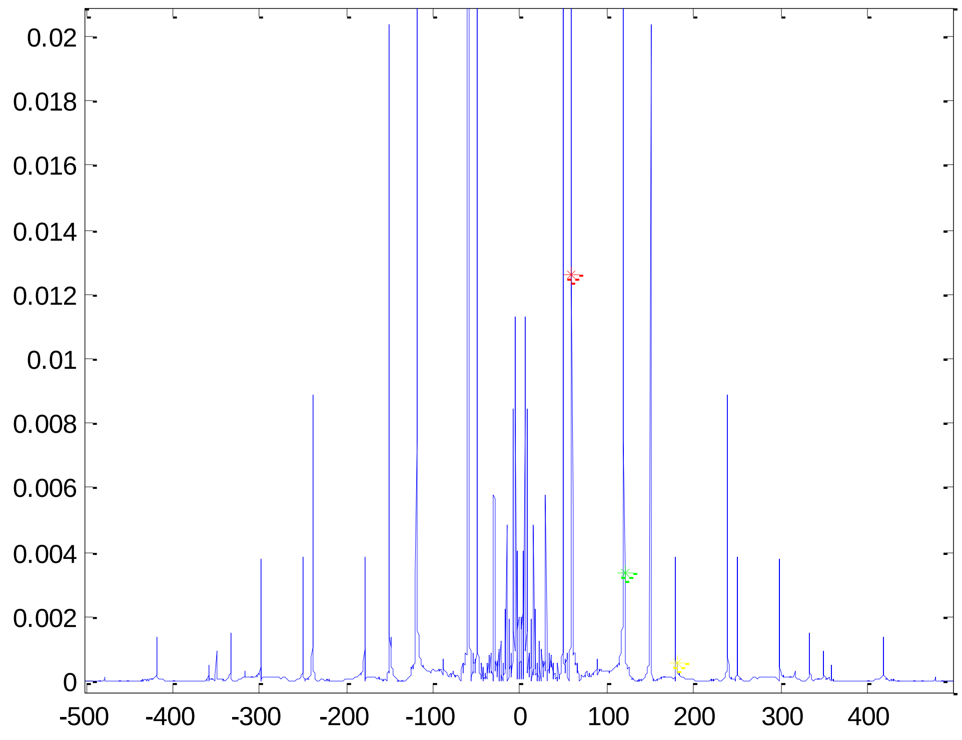

Fig. 4.23 noise's DEP

Chapter 5

Modulator

In this chapter, I'm going to show the features that has the modulator for this thesis. In this case the modulator make the same as the normal modulation, however I add two functions more one is the function of divide the screen in some part in order to find the position of the token, and the other is also assign one orthogonal signal in all the neighbourhood boxes.

Another extra function is to synchronize the signal to send with the refresh rate of the LCD display, But this thing is made by the Java frame [22] working with full screen so we only use the Java features for send a new image every time when the LCD refresh the image that is shown. This is not the unique reason to choose Java platform, another important one is the possibility of using the programs written in Java in a multitude of computer Platforms.

5.1 Screen division

This part is really working with geometry, and the main point here is to find some type of grid, that when we apply it to one game board, and we place on it one token we can have to possible situations this are shown in the next figures.

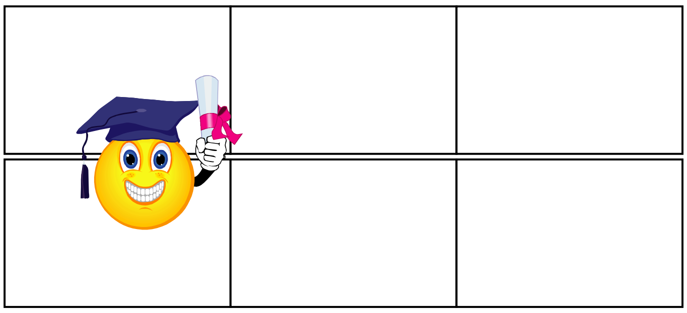

Fig. 5.1 Token in the grid

In the above figure the token is well placed in the middle of the grid, so it only will receive one signal coming from the board. However, the player can put the token not in the middle of one box, it can be placed in other bad places that show the figure, You can see in the next figure that the token on the left is placed above two boxes so it will receive the signal that comes from two boxes so it will be a problem, The second token in the right side is placed, but really he has a worst position that the first token, because it will receive the contributions of the four signals that are coming from the four cells that it has below of it.

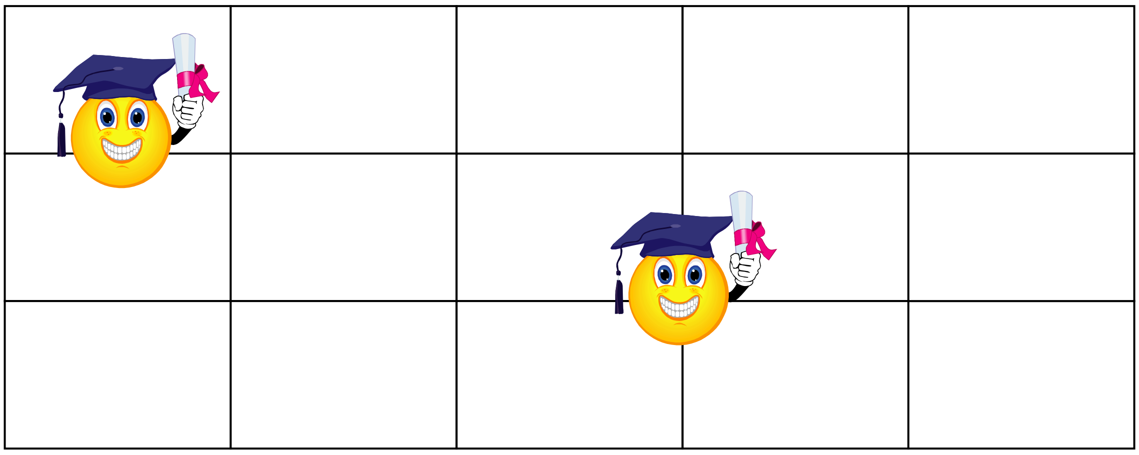

Fig. 5.2 tokens in the board

That means that will be good for us to find the structure that doesn't matter where the token is placed but it generates the less number of interference signals. This case is well solved in the mobile communications and the hexagonal grid, if for the division of the board I use and hexagonal grid doesn't matter the place that I put the token it can only see one maximum of three different signals.

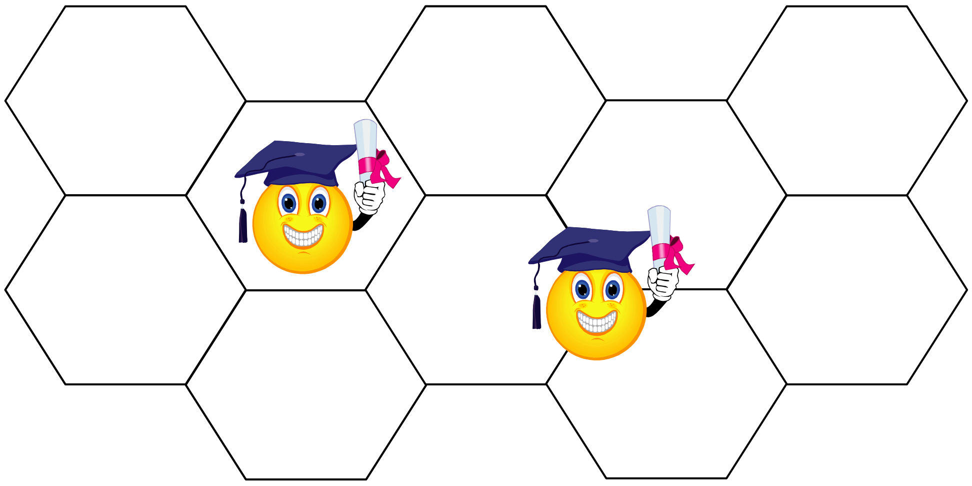

Fig. 5.3 Token in one hexagonal grid

So this is the main reason for the use one hexagonal grid. In this case we only need three different orthogonal signal in order to separate the three contributions that can receive the token.

But what happens if we cannot use this geometry and we need to use square boxes, or another distributions which is more irregular, the problem now is how many orthogonal codes we need in order to realize the task that doesn't matter we put the token it can separate all the contributions, in order to separate this contributions we need that every contributions must be an orthogonal to the others, so at the first sight of this problems we can think that we need a lot of orthogonal codes. But that is not necessary; we can use the result of one famous theorem. The 4 colours theorem, this theorem demonstrates that doesn't matter how irregular is one map, if we have four colours we can paint it and separate all the areas. So we only need four orthogonal codes for each map, then if we have four orthogonal signals, we can distribute it on the grid and we can accomplish that will be a different orthogonal signal in every boxes that are placed together.

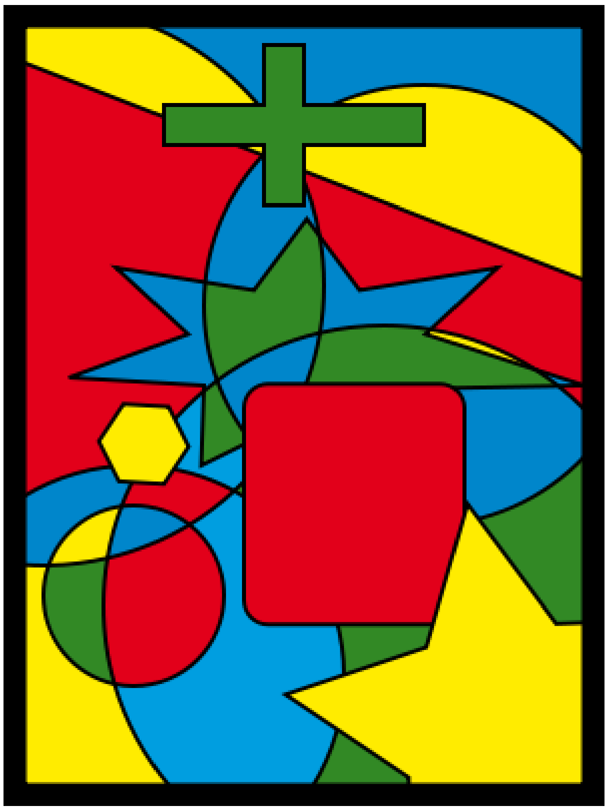

Fig. 5.4 Four colors

You can apreciatte it on the above figure.

Finally when we have divided the screen the only thing that have to make is send to each division one different code and then if one toke receive one code, is inmediate to know which position sends that code.

5.2 Add the signal to the picture

For adding the signal to the static image, we are going to use the refresh ratio of the LCD, at this frequency the display repaint the picture, so the thing that I¨m going to change from one frame to the other the brightness of the picture,i.e. if I have one pixel in RGB that have the values (127,127,127) which represents the gray color, in the next frame if I want to send a +1. I will send (128,128,128) oder for a -1 (126,126,126), if this change is sufficient small the human eye cannot see that. I we have the screen divided by hexagons in each hexagon I send one orthogonal code. For the next example I will use the Walsh code of length four, which is orthogonal, I will code the +1 with white and the -1 with black color, this make sense because the sensor receive changes of light and if it has white light will receive positive values and for the black will receive zero or negative values. But really we can codify the symbols als we want. In the next figures , we are going to show the four different images that we need to load in the LCd the four szmbols of the four walsh codes. First take a look on walsh codes.

$$\begin{array}{l} wo = +1 \quad +1 \quad +1 \quad +1 \\ w1 = +1 \quad -1 \quad +1 \quad -1 \\ w2 = +1 \quad +1 \quad -1 \quad -1 \\ w3 = +1 \quad -1 \quad -1 \quad +1 \end{array}$$

First divide the screen and assign the walsh codes

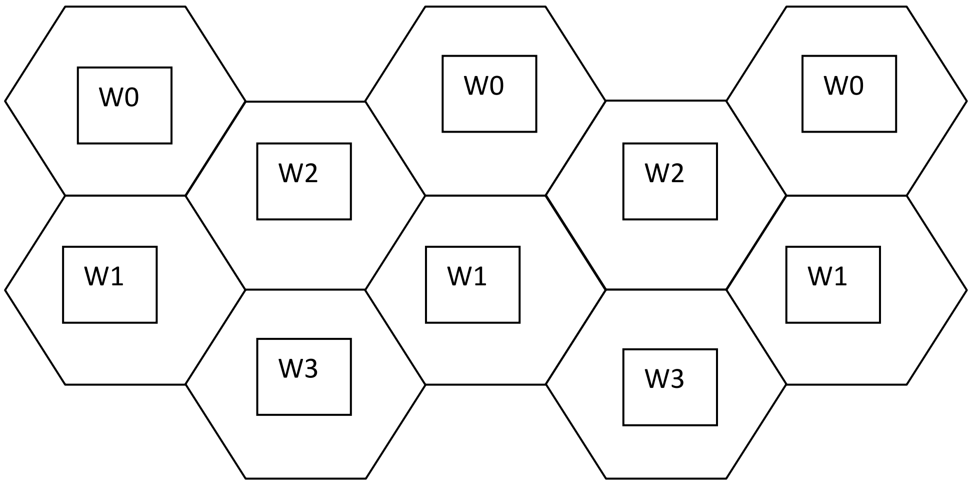

Fig. 5.5 Screen division and Walsh code assign

Then the first frame that will show the screen is this, you can see that this all hexagons are white corresponding with the +1. Let's also put one token in one position

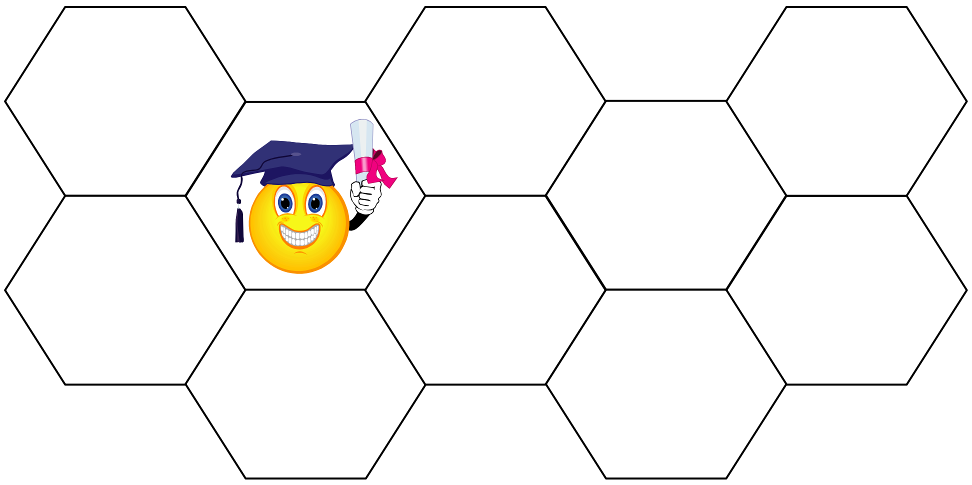

Fig. 5.6 Frame corresponding Walsh code symbol 1

Let's go for the next symbol, now there are two hexagons that has to show one -1

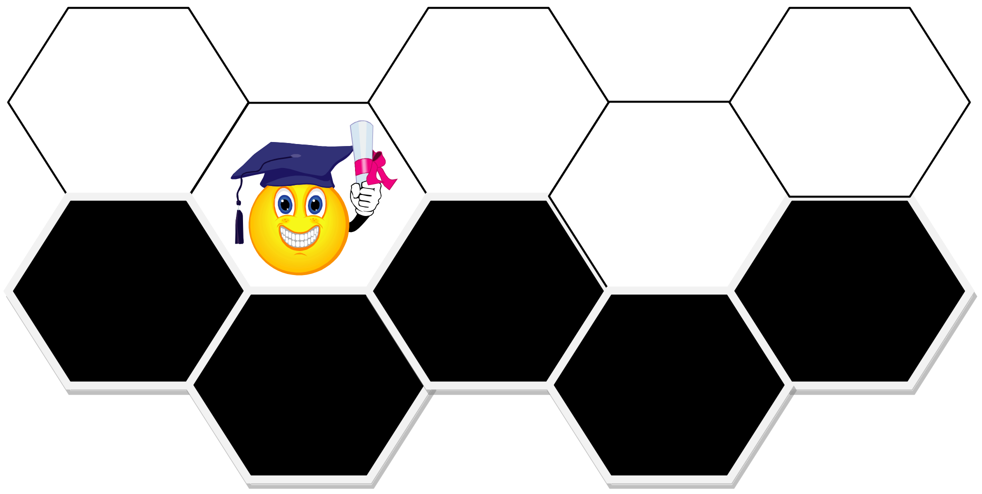
Fig. 5.7 second symbol

Then the next symbol is the third one

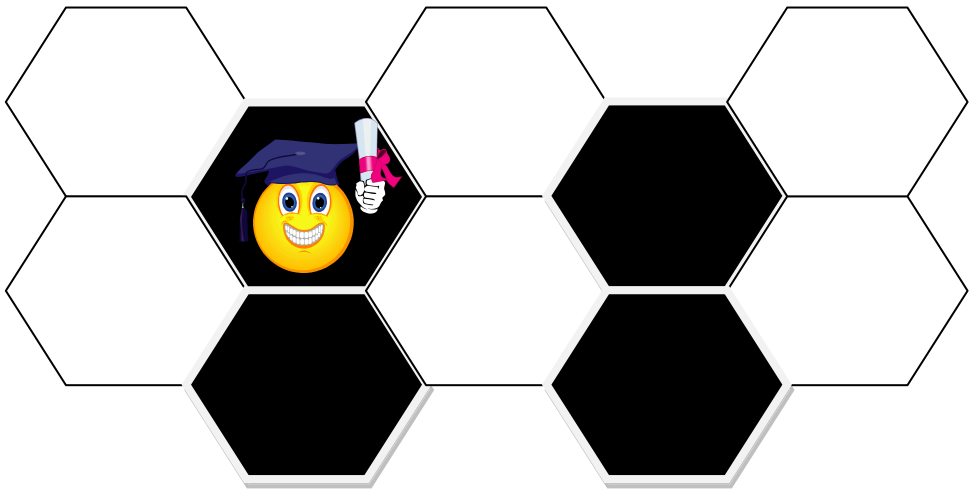
Fig. 5.8 third symbol

And finally the last symbol:

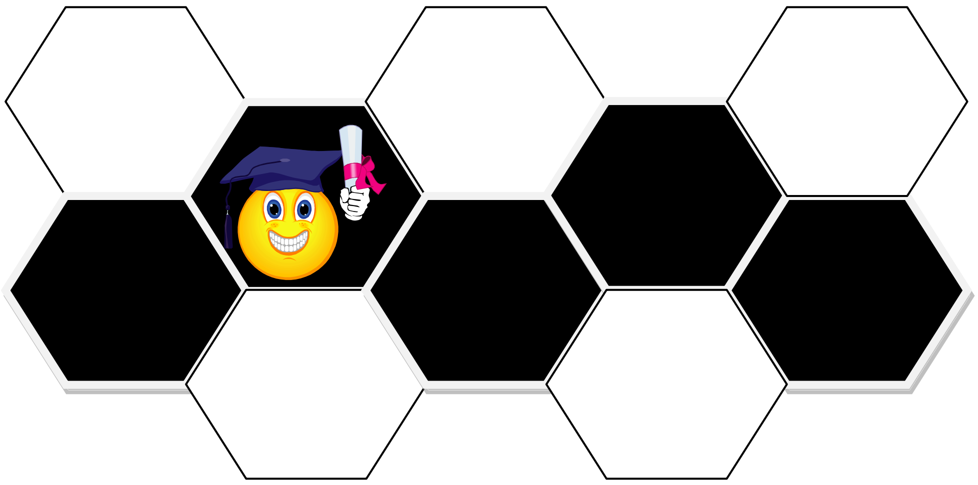

Fig. 5.9 fourth symbol

And then the LCD displays the images but one after the other, now we can find which Walsh code has seen the token. You can obtain this sequence: white, white, black, black. Which correspond to +1, +1,-1,-1. And this is the w2 so we can find that the token was in one cell that sends the w2.

Obviously if we move from the black value in RGB (0,0,0) to the white value (255,255,255), the player can see that, but what happens if only change the brightness of one hexagon only just a little, i.e. suppose that the hexagon has this colour in each pixel that contains (123,124,153) only change to (124,125,154) for the positive value and I do not modify the RGB values for the negative symbol, the user do not can see, that in Appendix A the reader can see the result of one test that I made to my partners here. In this moment is a good movement to discard from using PAM modulations because the human eye cannot see a little change of only one level, from for example 124 to 126, but when the signal is moving if we change more than three at this low frequencies the human eye can see the flicker.

To assign the Walsh codes to the hexagonal structure, in order to avoid the problem of having the same Walsh code in two consecutive cells we can use the distribution algorithms described in [24], [25] and [26].

We use Walsh codes because they are orthogonal, however we can use sequences that have good correlation properties, in order to know if we receive one or more sequences.

Another thing is to use Pseudo noise codes, PN however this sequence need to be long in order to achieve good cross correlation properties, so in our case it's not good because we send only one symbol every time that the LCD change the displayed image at this frequency is very slow. So that is the reason that I only use orthogonal codes.

Finally we have all the things to form out the frames, the only thing is define the header of this frame, and this will be explained in the nest section.

5.3 Synchronism

First I'm going to justify what is important to add synchronism to the frames that we are going to send, and then I will explain the two types of synchronism that the system can use.

One of the two reasons of add synchronism is because, some orthogonal sequences if you make the permutation of one of them we have another orthogonal sequence. So we cannot know which sequence the system send. I.e. If we use the Walsh code ++-- and we send always this code, when we put the token to catch this signal we can be unlucky and put the token in the second positive value and receive this +--++--+ which we can identify with the other Walsh code that is +--+. So we need to put one bit to not have this effect.

The second reason of use some frame for Synchronism is because of, when the system send more complex sequences, like combinations of Walsh codes we need to know which one is the first sequence, so for that our frames need synchronism.

I define two types of synchronism for this transmission:

- Use a different level
- Use barker codes

The first is easy, I will use only one bit of different level, for this my frame using Walsh codes will be something similar to this -2,+1,+1,-1,-1,-2,+1,+1… In order to not loose orthogonality all codes will have this synchronism even if they do not need.

The second is to use one barker code, this codes are used normally in frame synchronism in this case I will use the barker code of length four and the one with length five.

5.4 Modulation BPSK or PAM

Now we have the Walsh codes and the synchronism, so we can assign one Walsh code to each cell and send it, decode and find the cell that it's placed the token, the problem is that we can only use four cells, so we need or more Walsh codes, more orthogonal codes or use another technique.

We can extend the Walsh codes using a Hadamard matrix, and we can have the number of Walsh codes that we need, the problem of that is that if I need 8 Walsh codes their length must be 8, and if I need 16 Walsh codes the length is 16, Walsh codes always form a square matrix of length 2^n with 2^n codes inside. And for orthogonal codes occurs the same if we need seven orthogonal codes their length must be seven, this is algebra, if we want n codes that are orthogonal the length of this codes is n. So we need another thing that this in order that allows having mode cells than Walsh codes or orthogonal sequences we have.

In order to make that we need this. If we have two orthogonal codes and multiply one with one scalar the result is also orthogonal to the other sequence, so we can multiply our sequence by (-1 or +1) and have some case of BPSK, modulation.

Finally, I have all the things that I need for codify the different boxes in the game board.

5.5 Building the frame

I use two techniques for building this frames, but really the reader can make it as you want.

The first case is that always the box send the same orthogonal code and modulate it with the his position in the LCD in binary. In order to do this, I will show this example.

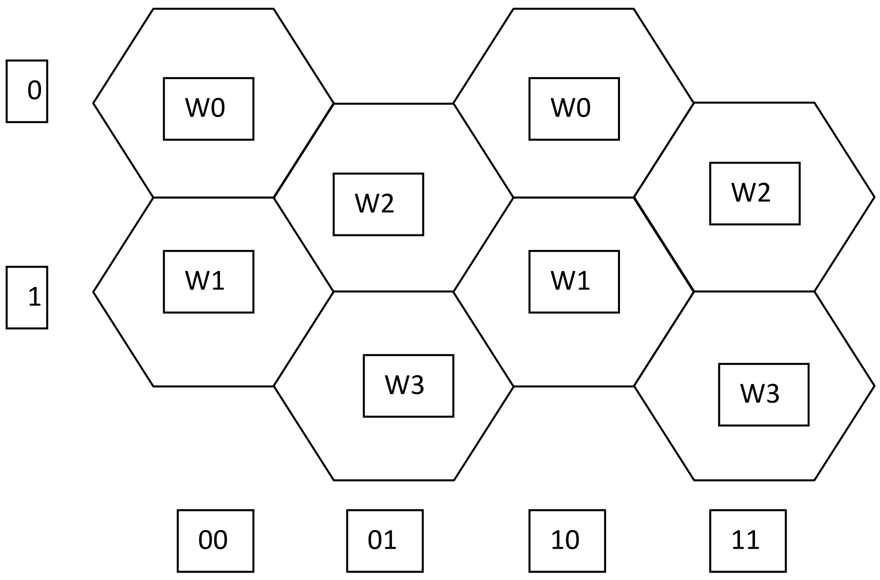

Fig. 5.10 cell assignment

First of all, you must code the cells with one unique number in this case, the code is to put one number in each cell an row, so we can call each cell with the combination of two numbers, like a matrix,

The next step is to put one orthogonal code, in this case Walsh codes, on each cell, named W0, W1,W2;W3,

When we have this things defined, the frame of the system will be the synchronism and then the Number of Walsh codes that the system need to code the positions. Due to make a good reception if we do not the code, the length of the word must be the same as the code bit, so we need three Walsh codes, one for code the number of the row, and the others two for the coding of the number of the column which the cell belongs. So the four possible frames in this moment will look like this:

Sync	W0	W0	W0

Sync	W2	W2	W2

Sync	W1	W1	W1

Sync	W3	W3	W3

Fig. 5.11 sequences to send

When we have all this, we only must code the number of the row and columns, as I said the first word for the row and the other two for the columns, then the thing that must make the modulator is if the row is one cero do not make any change to the word, however if it's one of the orthogonal code will be multiplied by -1. And then the same thing which each binary digit that has every column.

Now we have the full sequence to send.

But in this moment, is a good moment for think that we lost a lot of bits with this codification, This problem is one problem well known in programming science, which is to use less bits for storage matrix or whatever we have. SO this is a problem for the science of data structures and algorithms to find the optimum codification for each distribution.

In this case the most easy and short codification is jointing all the four cells that has different Walsh codes and called it one station, so in Figure 5.9 we will have two station, station 0 and 1, and then if we code this we can have sequence with only one word to code all the cells.

Sync
W0
Sync
W1
Sync
W2
Sync
W3

Fig. 5.12 code using stations

And then multiply each Wx by the number of the stations that belongs, if is 0 multiply by 1 and if it's 1 multiply by -1.

With this example we illustrate how I code the frames and the importance of a good code of the positions. In this case if I use the same Walsh code the best codification is to use the second one, by stations.

The problem of using the same Walsh code for every frame is that we do not cover all the combinations that we can make if I use different Walsh code in each position in the frame. The advantage of this system is that we can detect easy the Walsh code that the system use, because we have more opportunities per frame to detect this, in the other case there will be only one opportunity per frame to detect it.

If I use the repetition of the same Walsh code, by combinatory I can only make four different sequences, and then I can apply the BPSK modulator so I will have this formula to know all the possible sequences if I use n Walsh codes

$$N = 4 * 2^n \ (5.1)$$

But if I use different Walsh code and also the BPSK the new number of possible sequences is

$$N = 4^n * 2^n \ (5.2)$$

It's easy to see that the second equation grows faster than the first. The first factor in this equations is the number of available orthogonal codes that we have, and the second factor is added by the effect of the BPSK modulation.

In this case I only use four orthogonal codes, but really this number can be bigger, the problem is that when this number is big we need more bits to represent the orthogonal sequence, as I said if I have 8 orthogonal codes I need 8 bits to code it, in generally if I have n orthogonal codes I need n bits to code it. So it's important to find the good compromise with this things. I work all the time with 4 orthogonal sequences but I try with longer codes and the system also works.

To code the position using this type, I make the same thing as before, I code the positions and then translate it to my system of coding the positions as I made before.

For example if I have one matrix 8x8 I can assign one Walsh code and the his modulated code to each column and row, and then send this information in every box inside the matrix.

	W0	W1	W2	W3	-1*W0	-1*W1	-1*W2	-1*W3
W0	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
W1	🟩	🟩	🟩	🟩	🟥	🟩	🟩	🟩
W2	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
W3	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
-1*W0	🟩	🟩	🟦	🟩	🟩	🟩	🟩	🟩
-1*W1	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
-1*W2	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
-1*W3	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

Fig. 5.13 code with different Walsh code in a frame.

For example to code the red and blue box in the above figure I will send:

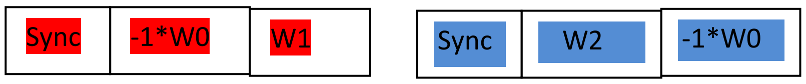

Fig. 5.14 Frames for two positions

Note that in this case with two codes we can codify 64 positions; if we use the first system I need 4 codes to codify the same number.

Finally the last thing that we must make is code the frame to send, for this the only thing that we must make is the code the orthogonal code that we use to his representations in numbers.

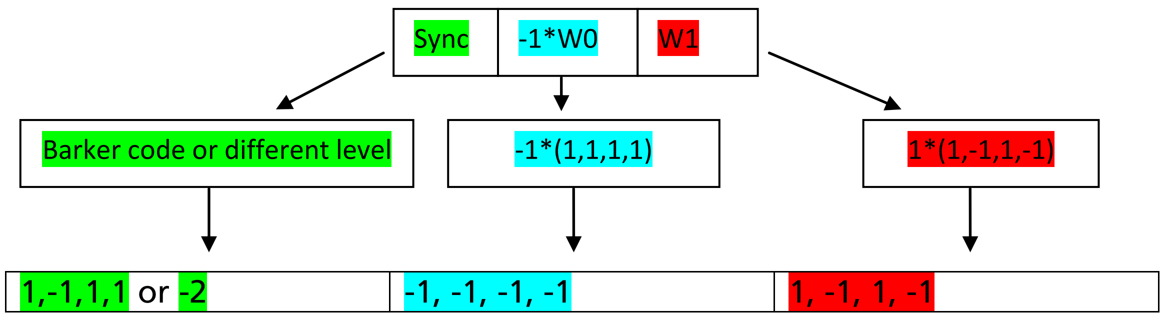

Fig. 5.15 frame conversion

In the above picture we can see the sequence that we are going to put in the screen for each pixel that belongs to re region that is coded by this sequence.

So let's see the different colours that will have one pixel that belongs to this area. I codify the -2 with black, -1 with gray and 1 with white

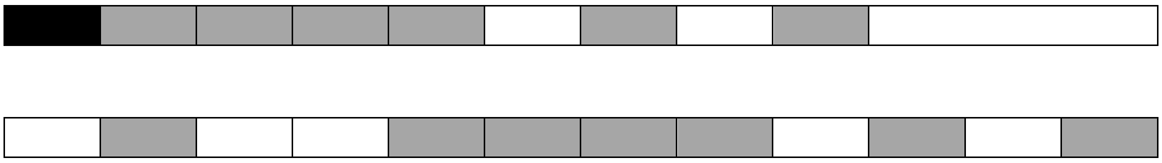

Fig. 5.16 sequence of colour to send to the screen

Really this sequence will be visible by the user, but if we only change the brightness a little the player cannot see anything. Finally you should notice that the first frame only has 9 bits and the second has 12 bits.

5.6 Programming the modulator

Like I said, I use Java for send this sequence to the screen and change the brightness of the pixels, I use two auxiliary Java classes, that their function is to divide the picture, which is shown in the screen and then apply to the RGB values of the pixels of this region, the sequence that this section has, and repeat it forever.

This image, that change every refresh rate thanks to Java frame Synchronization [22] at full screen. This programs are in appendix B.

Finally I need one third class that creates the sequence for each region, based on the position that will have every section I will assign one different sequence to each region and send this, to each region, the flow diagram of this program is shown in the next page figure.

For the assign of orthogonal codes of the hexagonal neighbours, we can use the algorithms that assign frequencies in the mobile cells. The idea is the same.

Program	LCD

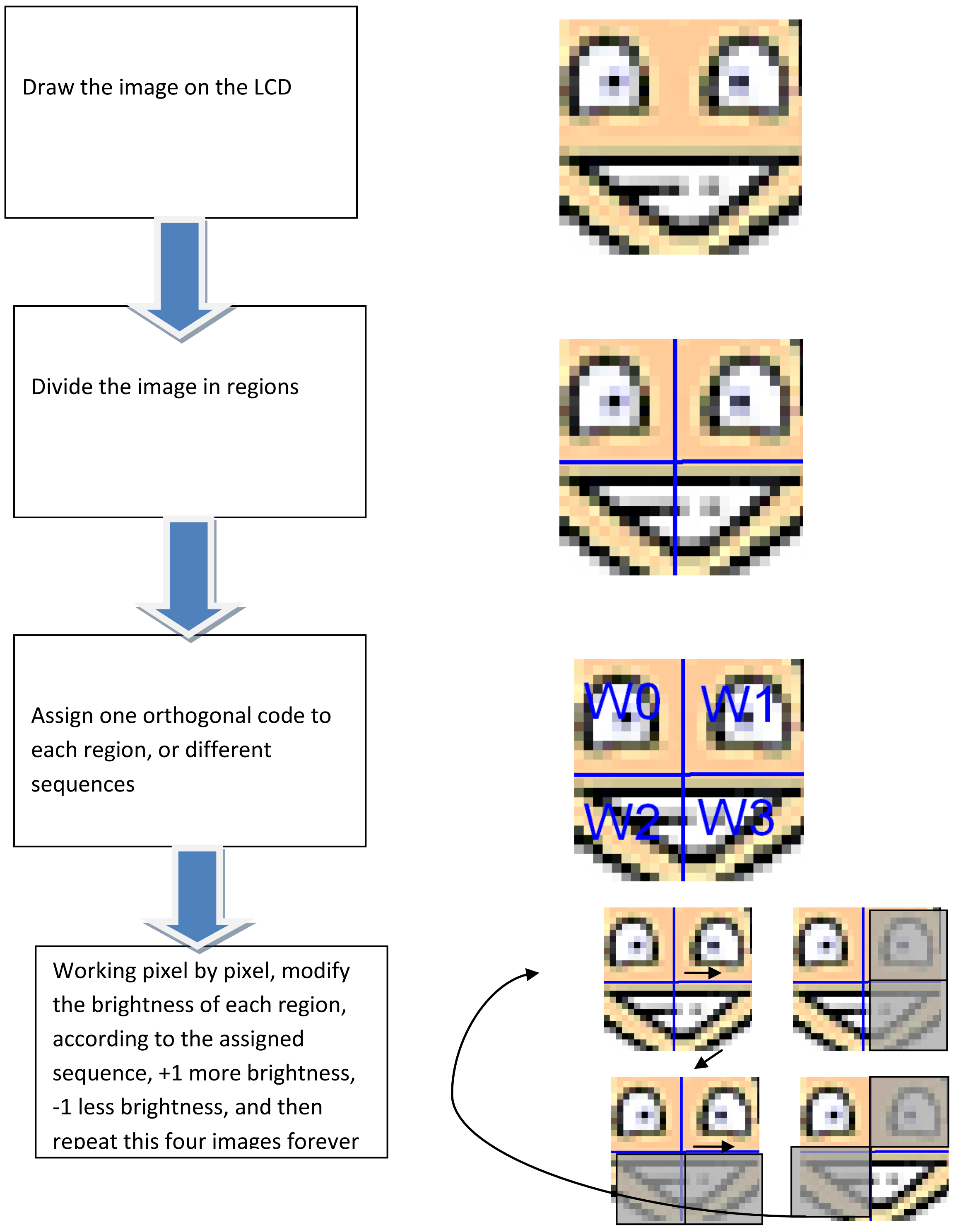
Draw the image on the LCD
Divide the image in regions
Assign one orthogonal code to each region, or different sequences
W0
W1
W2
W3
Working pixel by pixel, modify the brightness of each region, according to the assigned sequence, +1 more brightness, -1 less brightness, and then repeat this four images forever

Fig. 5.17 program flow

In order to make a good assignment of the sequence we have to be sure that every neighbour will have different Walsh codes. Then I construct the sequence.

Finally I will show how the picture and the divided screen looks like

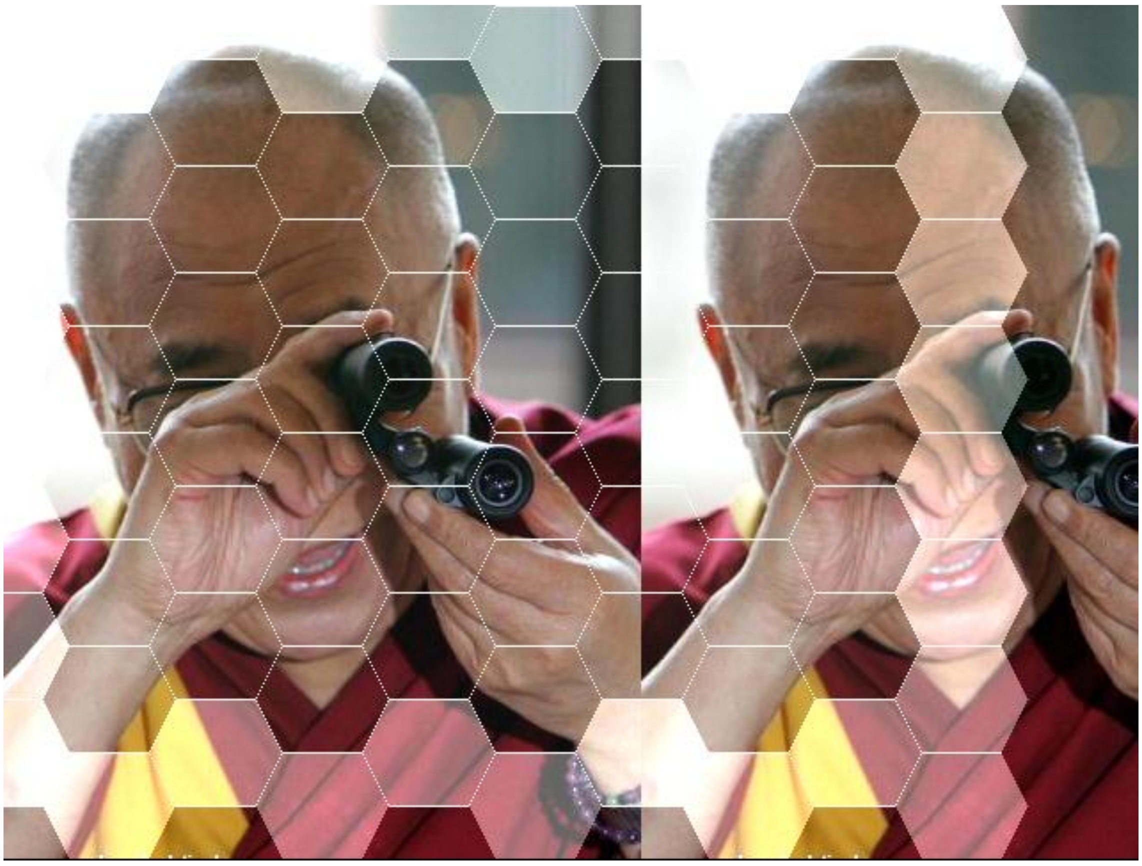

Fig. 5.18 picture in the LCD

Chapter 6

Demodulator

In this chapter we are going to talk about the demodulator features. This is all made in Mat lab, but I think that it will be easy to implement all this functions in a FPGA or PIC, the literature is well documented on the construction of an all digital receiver for demodulating signals. I think that this is the correct choice for now on to the future, it has a lot of advantages versus the analogical implantation, and in [7] we can find this advantages. This systems allow for building software configurable systems and provide better stability and reproducibility. So if we want to construct one demodulator that has flexibility, for example changing the symbol duration, in a software receiver we, uniquely, have to change the symbol rate variable and the next things will be the same, loading of course the new filters or what we need. So that is the case that we will implement an all digital receiver. In the following lines I will try to show you with results, figures and words how are working the different pieces of this demodulator

Our signal is a band base signal, the duration of symbol is imposed by the refreshment rate of the LCD, in our case 60 Hz, from now on fsy (symbol frequency), an our signal will be a BPSK signal but in some case we are going to have one additional level for the synchronization.

To get the signal we have the sensor directly connected to the audio device, then it samples the signal at 48 KHz, and use 16 bits per sample, then I apply the anti aliasing filter in this case the anti aliasing filter and the raised cosine filter is the same, the reason is easy why I need to filter the signal to times, but this implementation In the future will be made separately. Maybe the aliasing filter has to be implemented before the ADC, so it means in an analog way.

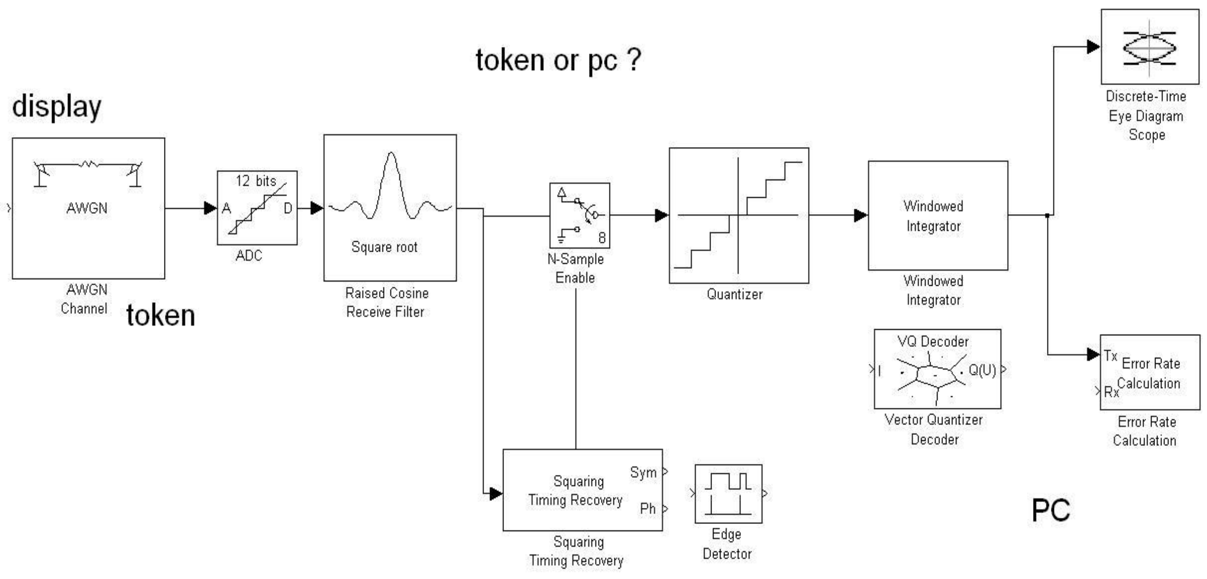

Fig. 6.1 Demodulator

6.1 Antialiasing filter

The problem of aliasing occurs when we have a band limited of bandwidth W, and then convert this signal to a digital one but sampling with a lower frequency than 2*W, this limit is called the Nyquist Frequency.

In our case our signal is a PAM digital signal with duration D=1/fsy, so it means by the equations of Fourier transforms that their bandwidth is at least fsy and some problems of ISI because his real spectrum is an infinite sinc.

The equations that define well this signal are

$$s(t) = \sum_{k=-\infty}^{\infty} a_k p(t - kD) \quad (6.1)$$

With a_k it the k-th symbol of the constellation of our signal and p(t) is the unit pulse of duration D, with (5.1) we can find their spectrum equation (5.2 shows this) if we suppose that symbols are uncorrelated in this case this is not true but for the analysis that we need we can figure out it.

$$G(f) = \frac{\sigma_a^2 \, \|P(f)\|^2}{D} + \frac{m_a}{D^2} \sum_{n=-\infty}^{\infty} \left\|P(\frac{n}{D})\right\|^2 + \delta(\frac{n}{D}) \quad (6.2)$$

$$P(f) = sinc(fD) \quad (6.3)$$

Let's plot the signals and show it for our case we have one pulse of width 16.7 ms. I set the amplitude of this pulse to one. Then I applied the Fourier transform to this function, the pulse is shown in6.1 and finally the spectrum is shown in 6.2 one sinc of 60 Hz.

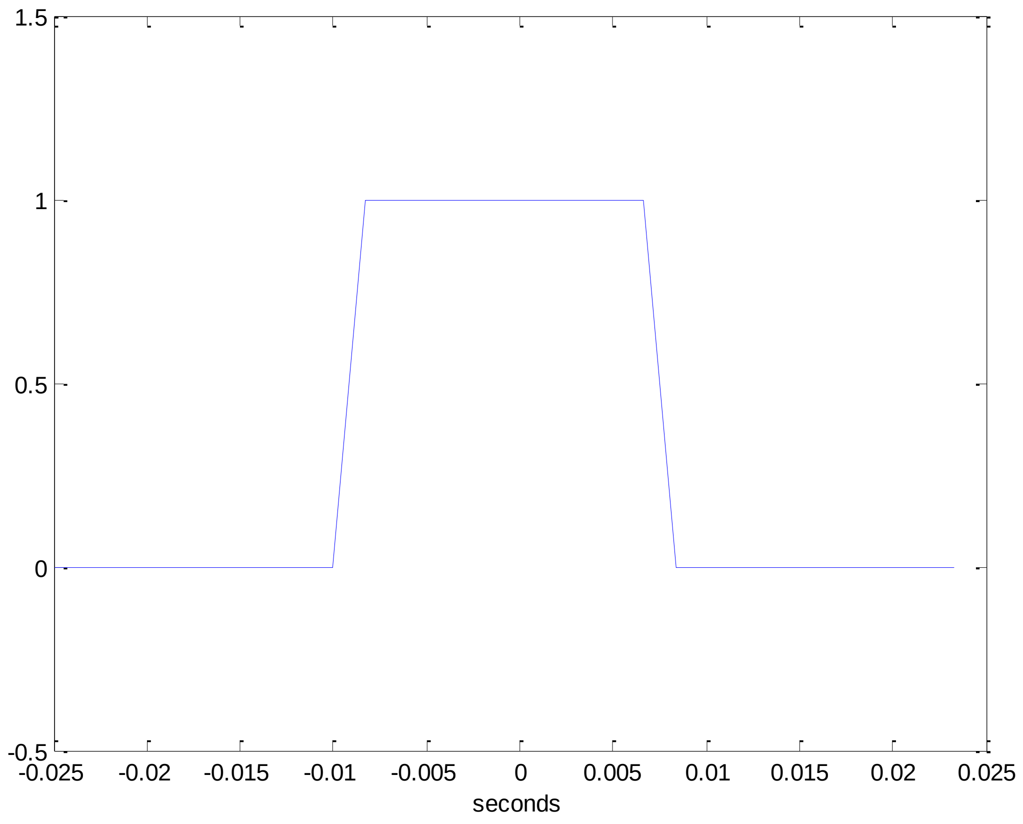

Fig. 6.2 unit pulse

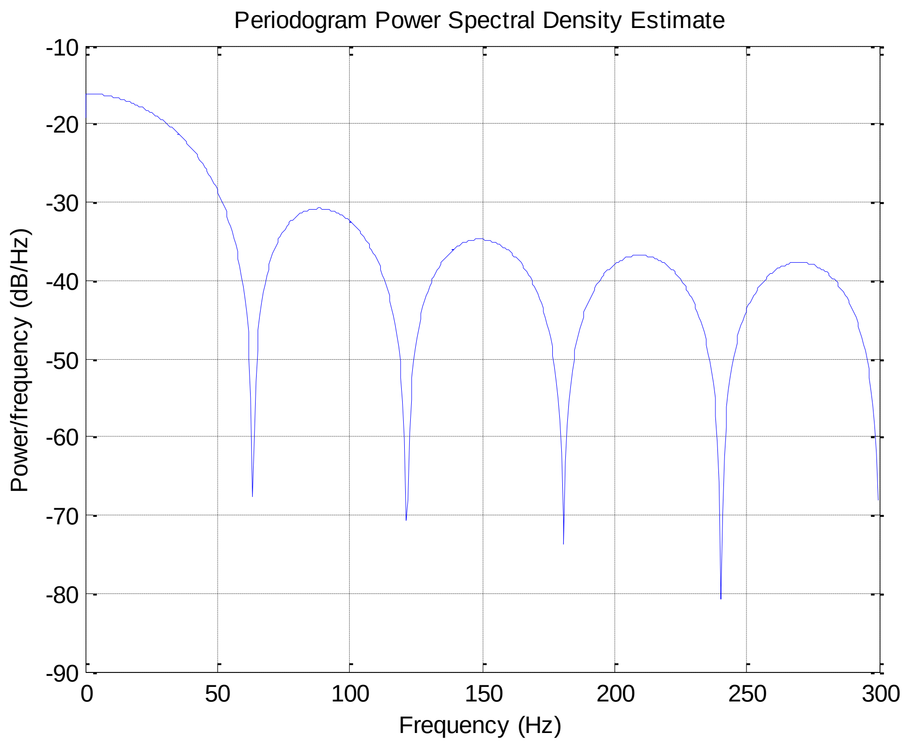

Fig. 6.3 spectrum our signal

So our signal will have a bandwidth near 60Hz, we notice that the minimum sample frequency using normal signal processing (use downsampling techniques for digital receivers is not necessary due the signal bandwidth is quite small) is 120 so our filter will have the cut frequency of 60 Hz in this case, in other case will be one half of the sampling frequency.

6.2 Raised Cosine filter

In digital communications, we can have the problem on intersymbol interference called ISI. This problem is due to the queues of the sinc that have the perfect digital pulse, the easiest way, to solve it, is to filter our signal with the raised cosine filter, and it also helps us to conform better the pulse that we have from the screen.

We also need a low pass filter for this receiver, so this is easy to implement, has the requirement of a low pass filter, with some good advantages:

- Good for recovery the symbol shape
- Can be implemented as a FIR filter with linear phase
- Easy implementation

This filter will be good not implement only at the minimum rate of sampling, it will be good make some upsampling of the signal (ten samples per symbol is enough) . this technique of upsampling before filter is sometimes used in some audio applications for having more precise filters.

Now I'm going to show the filter response in time and in frequency

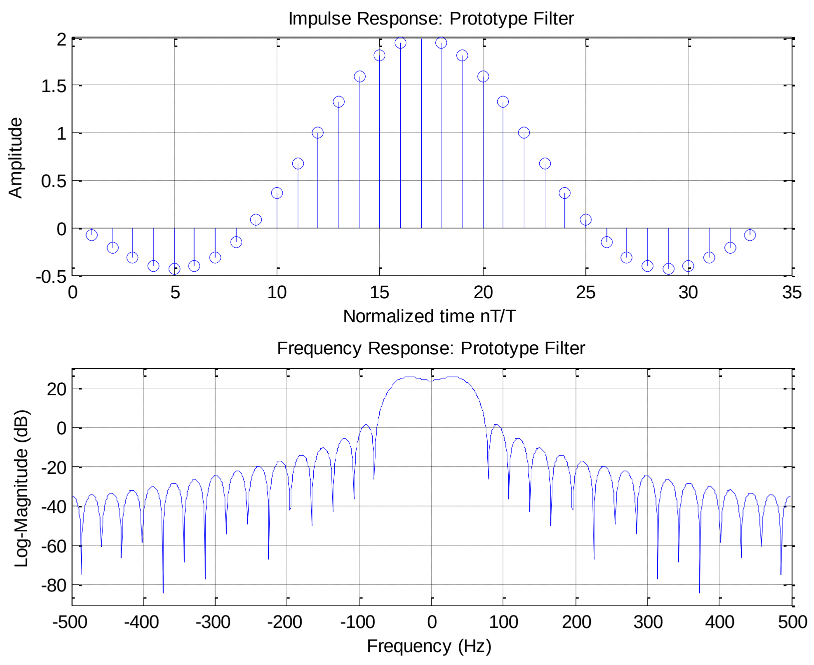

Fig. 6.4 raised cosine filter

In the above figure can see the coefficients of the FIR raised cosine filter, If we make a simple inspection of the first graphic, we can see that has linear phase, is symmetric, In the second figure, we can observe the Frequency response of our filter is similar to a low pass filters but in this case it let to pass some frequencies that allow to conform the pulse of symbol transmission.

It is also good for removing the noise, that produce the back light noise because this noise is positioned at 317 Hz in my display it also remove the normal AWGN noise that is added to the signal. Now in the next to figures the lector can appreciate the comparison between the signal captured and after that filtered by the raised cosine filter, the second one is the signal but now it has back light noise, we can observe that this noise is similar to AM modulation, so we make a Low pass filter in order to obtain the band base signal. Note that in the next figure we avoid the time during the signal has a transitory state, after this picture we can work with the minimum sample rate without problem.

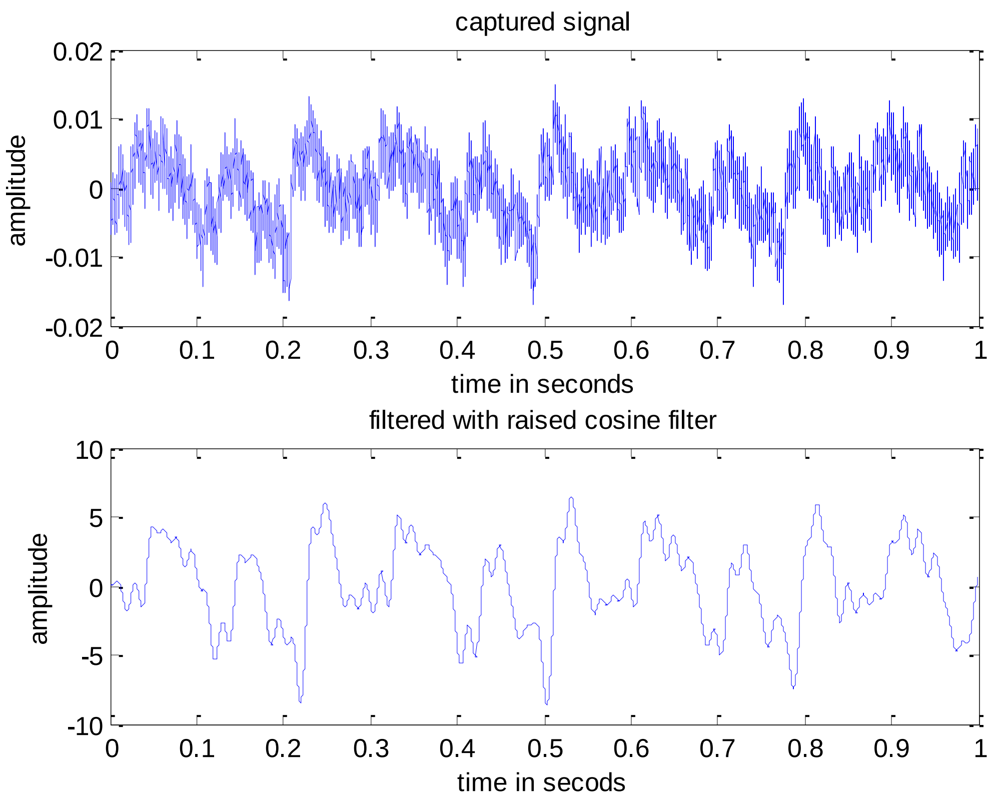

Fig. 6.5 signal without backlight noise filtered

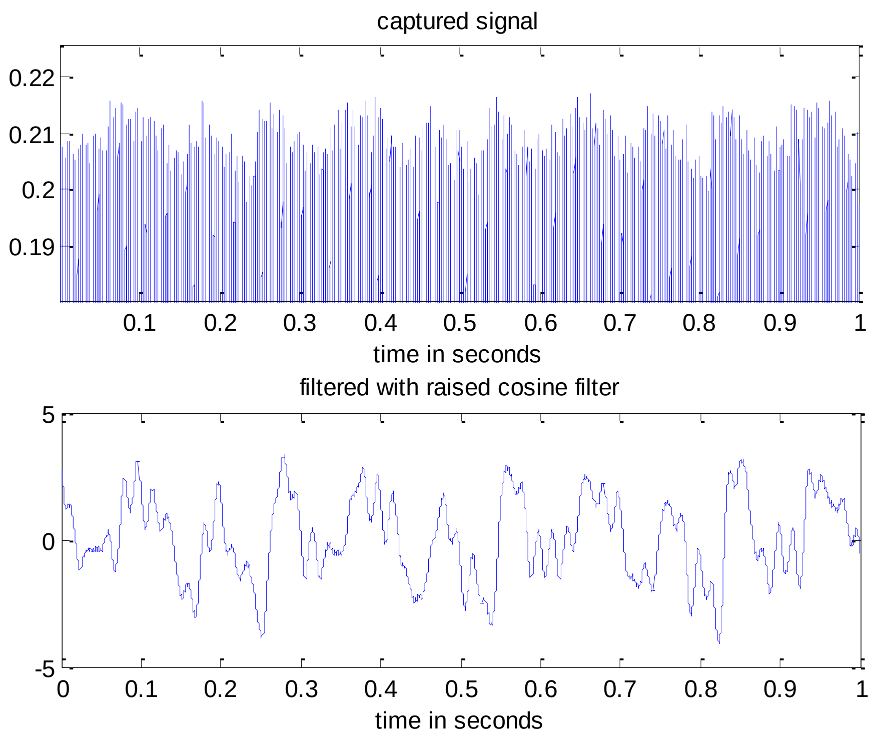

Fig. 6.6 signal with backlight noise filtered

6.3 Notch bank filter

For this signal, I believe that will be good to put three notch filters, and now I'm going to explain this reason for each filter.

A notch filter is one filter that eliminate only one frequency, this is often used in audio signals for eliminate tone interferences, or for example in electrocardiograms for eliminate the 50 Hz tone that comes from the power network.

6.3.1 Notch at 50 Hz

This is the first notch filter, I put this because of our signal frequency range is between 0 and 60 Hz so is very important not having this interference. It's easy to implement and avoid some problems in the future.

For this implementation I choose a second order IIR notch filter which is well defined and used for audio applications. The equations in a digital way for this notch are here:

$$h(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{a_0 + a_1 z^{-1} + a_2\, z^{-2}} \quad (6.4)$$

$$\begin{gathered} b_0 = 1 \\ b_1 = -2 * \cos(\omega) \quad (6.5\,) \\ b_2 = 1 \end{gathered}$$

$$\begin{gathered} a_0 = 1 \\ a_1 = -2 * \cos(\omega) \quad (6.6) \\ a_2 = 1 - \alpha \end{gathered}$$

$$\begin{gathered} \alpha = \frac{\sin(\omega)}{2Q} \\ Q = \frac{1}{fc} \qquad (6.7) \\ \omega = 2 * pi * fc/fsamplig \end{gathered}$$

I do not consider putting the notch around the multiples of this frequency because this part of this signal is filtered enough.

We are going to show the response in frequency of this notch filter

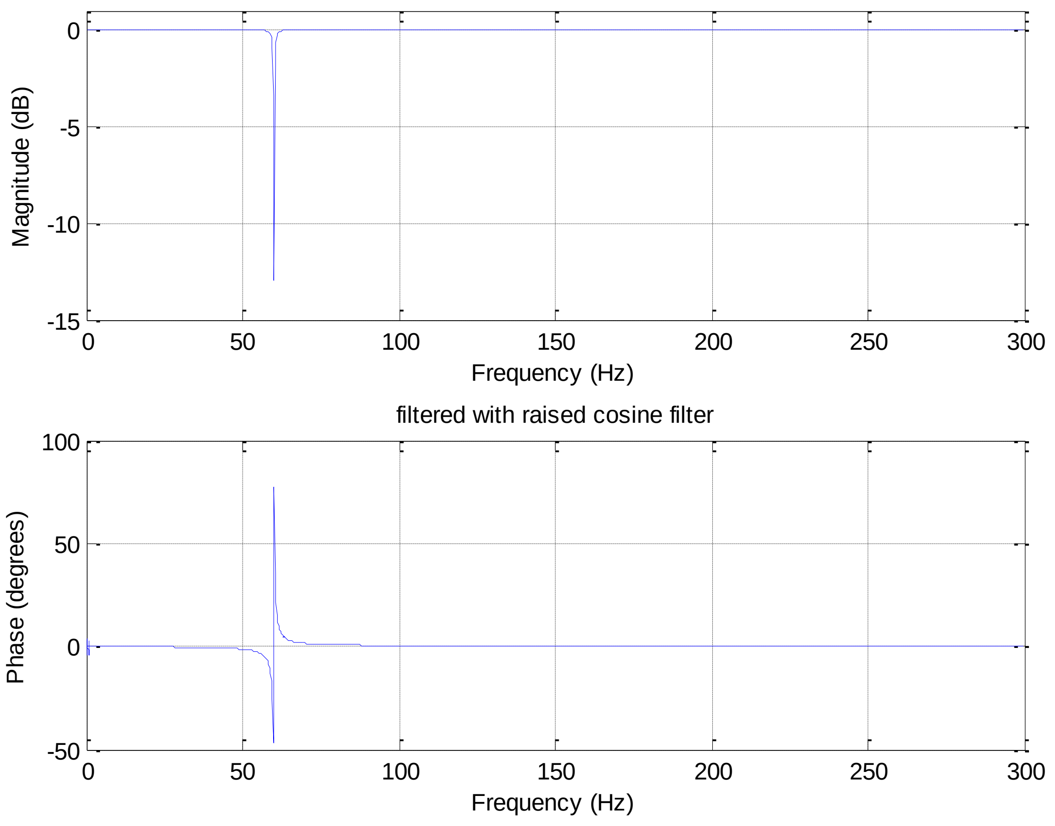

Fig. 6.7 Notch at 50Hz

6.3.2 Notch at 60 Hz

In the frequency of 60 Hz, this frequency is the refreshment frequency of the LCD, the signal always falls and the feeling is if we put some notch to this frequency our signal is more similar to one PAM signal, for the design of this filter we use the same equations describe in (6.4 to 6.7).

For this filter and the next we do not show the response because, do not supply any new information, the unique change is the pole, now it is in 60 Hz

6.3.3 Notch at the backlight noise

Despite we have made a low pass filter to our signal and remove this, we can appreciate a little undulation on the signal so, we still have this noise, we operate in the same form as before and put the notch filter to this signal.

To end with this part I'm going to show the frequency response of our signal with these filters and without them. So the filtered signal here is the signal filters by the raised cosine filter of precedent section and these three new filters.

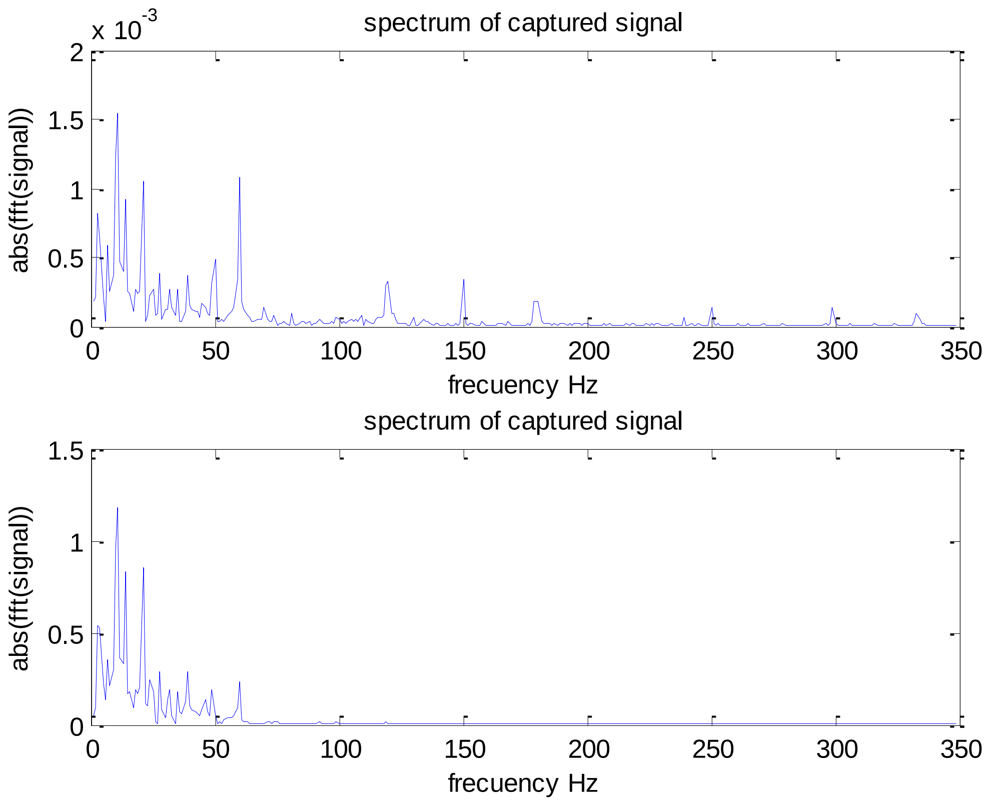

Fig. 6.8 spectrum of signal without back light noise

In the above picture we can compare the Fourier transform of the captured signal and the filtered signal, the second one only have the band with that we need, because the signal has been filtered, also we can see that has not any peak on 50 Hz and 60 Hz, because this peaks are filtered by the notch filters, at this frequencies we have the interference tone that comes from the power net and the tone that homes from the screen changes which for us are only noise.

Finally let's take a look on the signal if it has back light noise, on the next picture we can see that. The first drawing only has a delta near 320; this is the backlight noise that totally makes our signal, but then when we filter this signal, the spectrum of the resultant signal is equal to the signal that doesn't have back light noise.

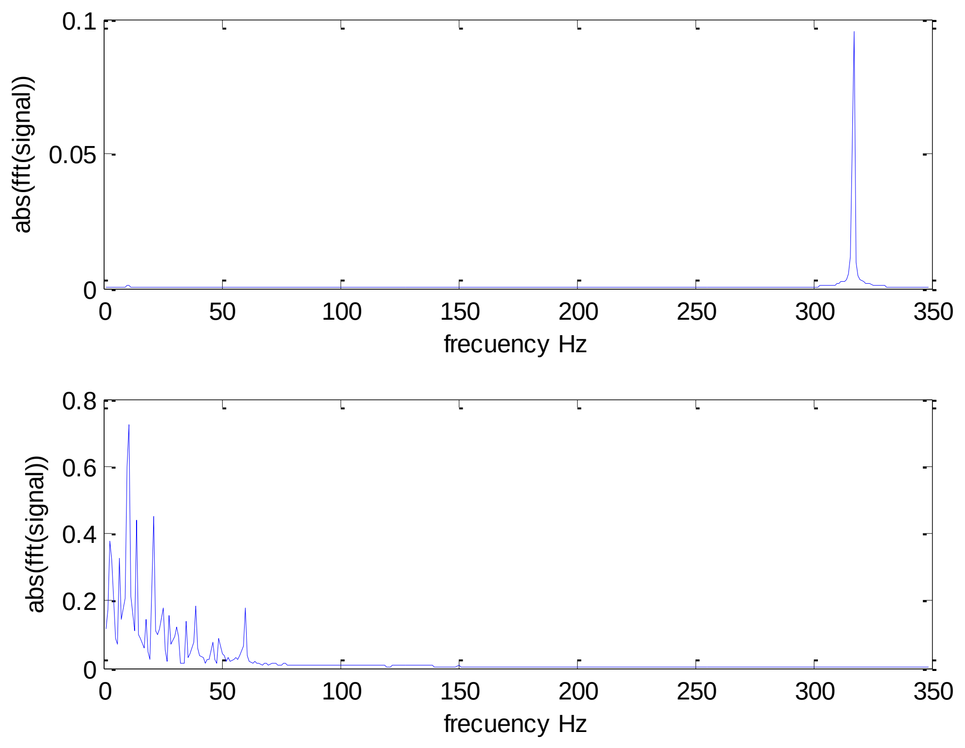

Fig. 6.9 Spectrum of signal with back light noise

6.4 Symbol timing recovery

When we have our signal and we have it filtered, the next step is to find the correct point for sampling. As we talked in the beginning of this chapter the receiver must be an all digital receiver so the symbol timing recovery must be done in a digital way.

In the modern days there are different technologies that can recover it, using this I will put in the literature all the papers that I read from this topic, I think that it's better than explain all of them. Here I'm only going to explain what I use for make that.

In the next picture I'm going to show the three methods for the symbol synchronization, this are the analogy recovery in this moment we cannot implement this we only have a sampled signal, the next one has the problem that is needed a back path to the analogical circuit that makes the sample to adjust de sampling frequency, I believe that this it's not a good solutions in this days. Finally the last is the current way to work now, only work with digital signals. It has all the advantages on digital processing, and sometimes for example in a polyphase filter bank we need more precision for working so the only thing that have to make is upsampling the signal and all is solved.

Another advantage of this is the possibility of easy reconfiguration of this system.

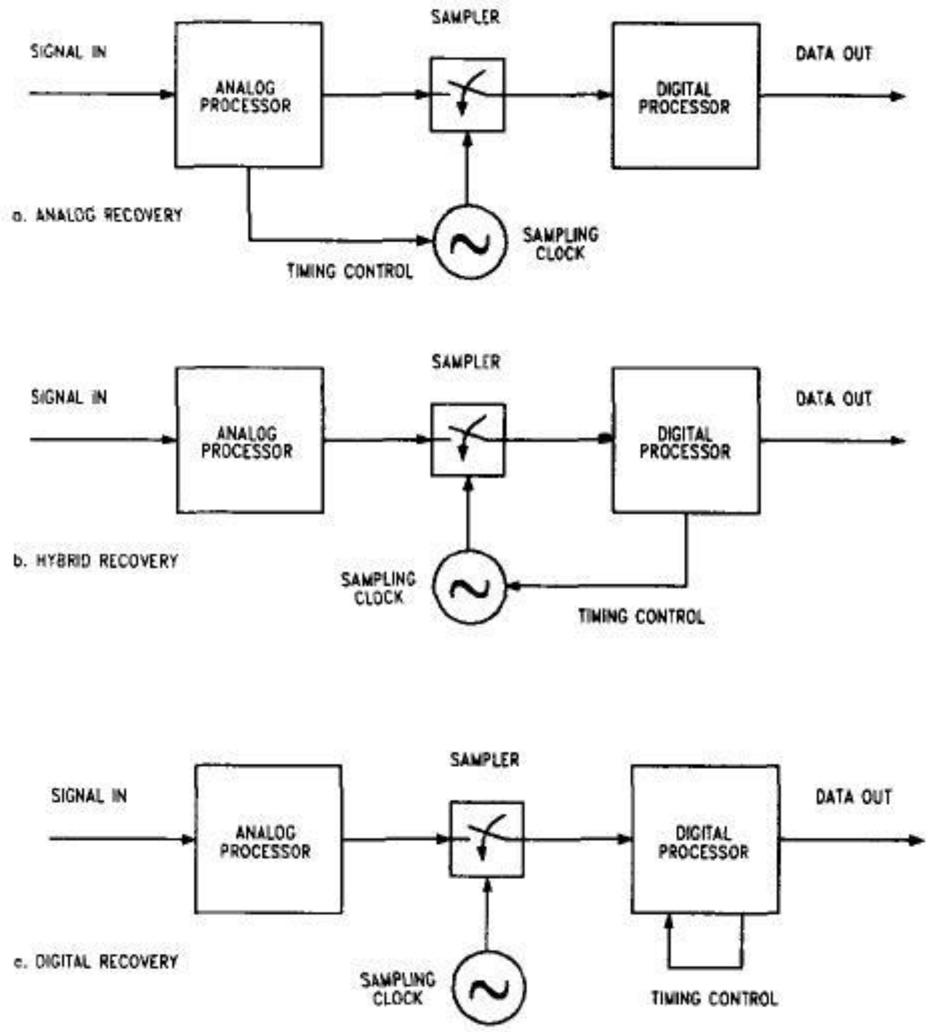

Fig. 6.10 different forms for timing recovery

For the moment I work with some different schematic, but this works quite good for this signal so I have already this implemented. The circuit that extract the clock and the timing error of our signal is this, after the picture I will explain how it works. There are a lot of techniques for the symbol timing recovery for example the Gardner Algorithm which is explained in [8] and a good implantation is made in [9] and [10].

In my case I have to use an open loop synchronizer because I haven't got any clock in the token for generate this signal and this is also easy to implement. First I get the signal and squared it then I have a Non return to zero code, this is in this way because the signal that we receive is quite smooth and not have big edges, then I filter this by a peak filter of order two centered at the symbol rate, Then I have some signal that the positive values are the correct point for sampling our signal, the blocks that make this are shown in the next figure.

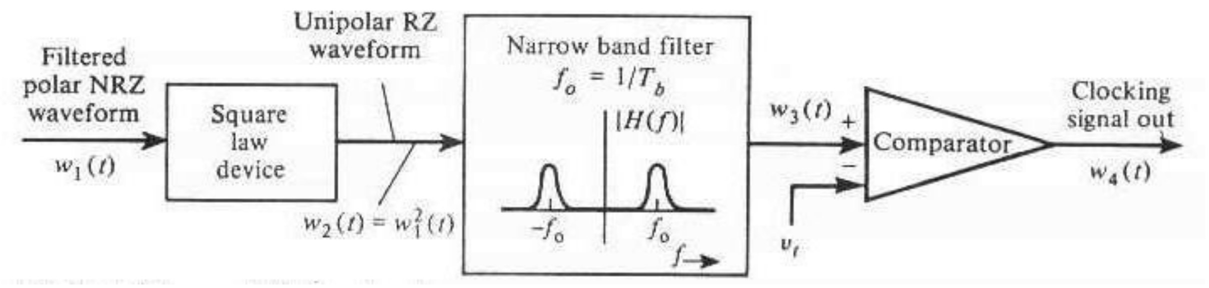

Fig. 6.11 Bit synchronizer

Another form to recovering this with works for all types of digital signal is using an differentiator for the timing error recovery, the problem of this is that the differentiator is very sensible to the high band noise so here recovers more importance a god low pass filter in the beginning, so I recommend to catch the signal at sufficient rate to not have aliasing, and after an antialiasing filter, and then make upsampling of the signal to filter that signal with a precise filter. The next figure shows the blocks that has this synchronizer.

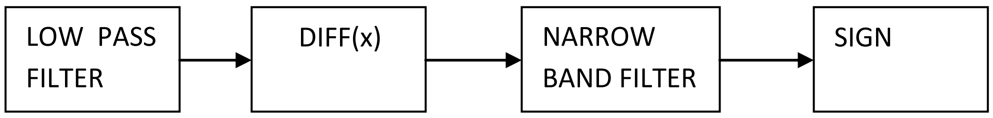

Fig. 6.12 bit synchronizer based on the

This loop operates at 10 samples per symbol, with this we can get a good compromise between hardware and BER.

Once we have well defined the timing error recovery we must find the correct sampling point in the digital way this is called interpolation.

The signal that I get from the output of these signal error detectors is shown in the next figure, in that figure you can appreciate the clock signal growing in the second window, and finally the third are the points that are good for the last sampling at symbol rate of our signal. So with this we can make two things the mean of this values or only get the middle values, once we are in this step really we cannot notice difference using one or the other technique.

The important thing here is only to find the desired samples in the next figure you can see the signal, and we take only the sample which is in the middle, and finally the last one who the symbols that we get from each period. The last signal is the signal sampled at 48 KHz and with a lot of back light noise.

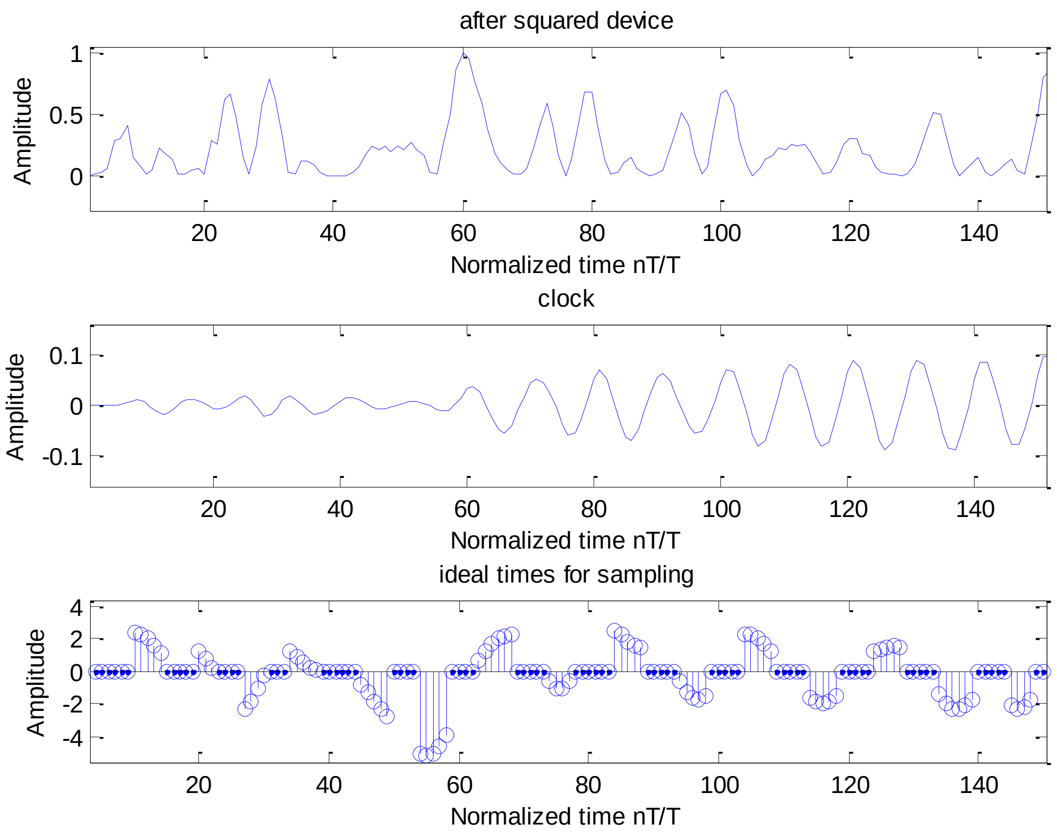

Fig. 6.13 Clock recovery.

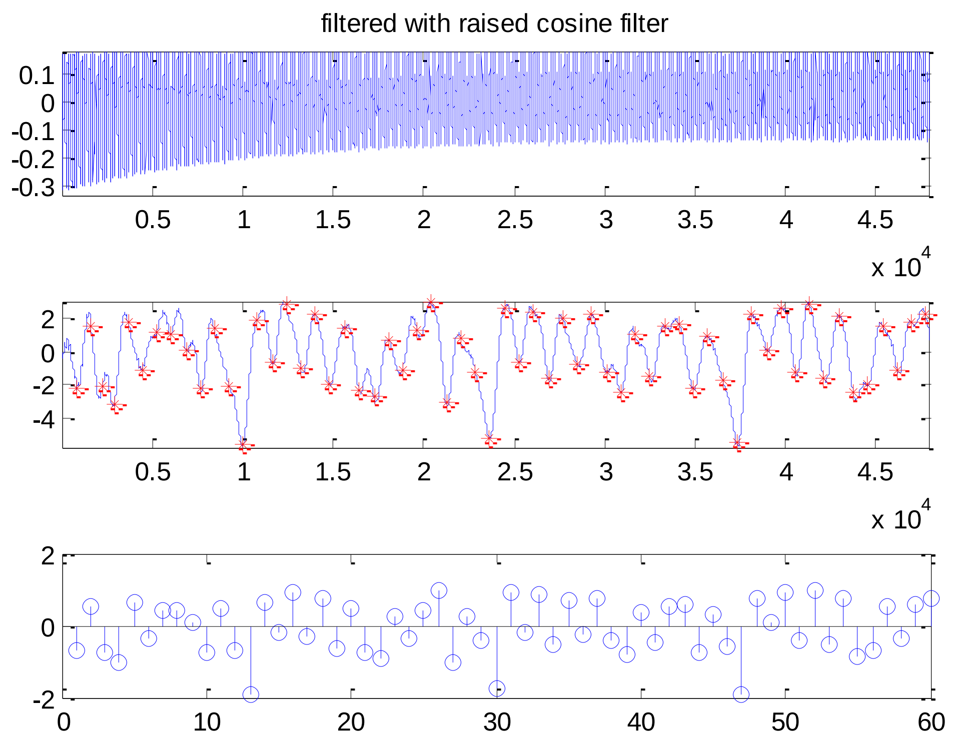

Fig. 6.14 symbol sampling

For the end of this part, I will show you the diagram of this symbol timing recovery and then some advice for the real implementation on a FPGA or a PIC. And then I Will show you the last to figures for this part the eye diagram and the constellation that we recover from one binary signal and one 3-PAM (Pulse amplitude modulation) digital modulation.

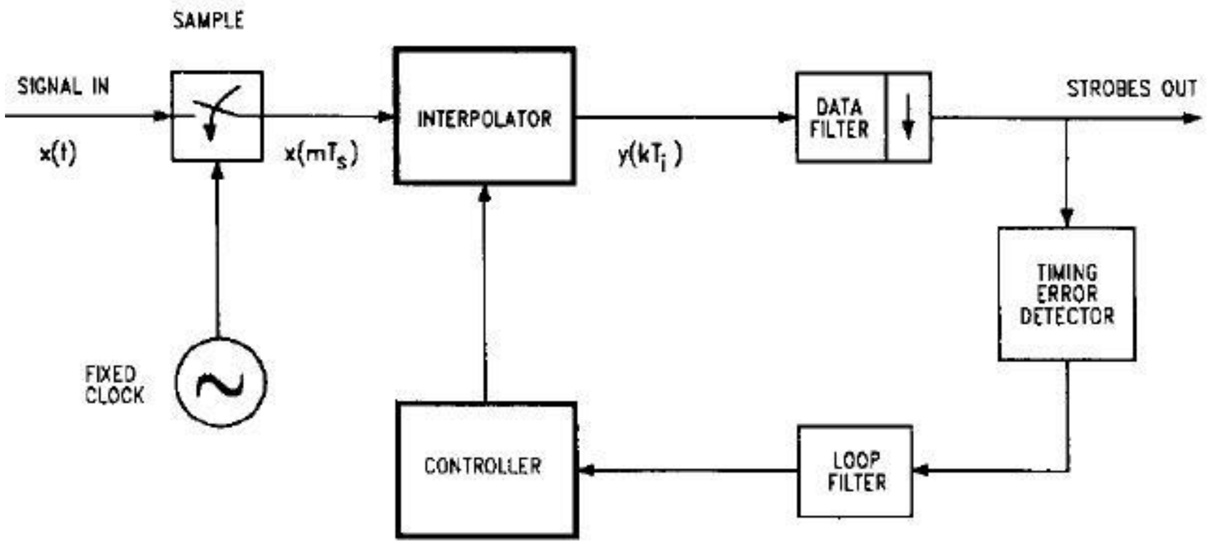

Fig. 6.15 Symbol timing recovery

In my case the timing error detector is placed immediately directly the interpolator but this doesn't affect it so much.

For the implantation in a FPGA or in a PIC. I will recommend using polyphase filter banks this are well described in [11] and [12] and some similar papers that man can find in the IEEE site and I will put in the bibliography.

Finally the last thing for all symbols timing recovery is to take a look for the eyediagram and the constellation.

First you are going to see this figures for the 3-PAM modulation, in this figure we can see three holes instead of only two, that hole is produced for non linearity in the screen when the signal is not enough random. But the demodulator can easy recover the three signal values. The problem is that the signal has a lot of excursion nut we can solve it with some adaptive filter, but really the correlator solve this.

The next figure is for a BPSK signal and we can see a very good eyediagram so know that we have well recovered the signal timing we can start with the next part of the demodulator.

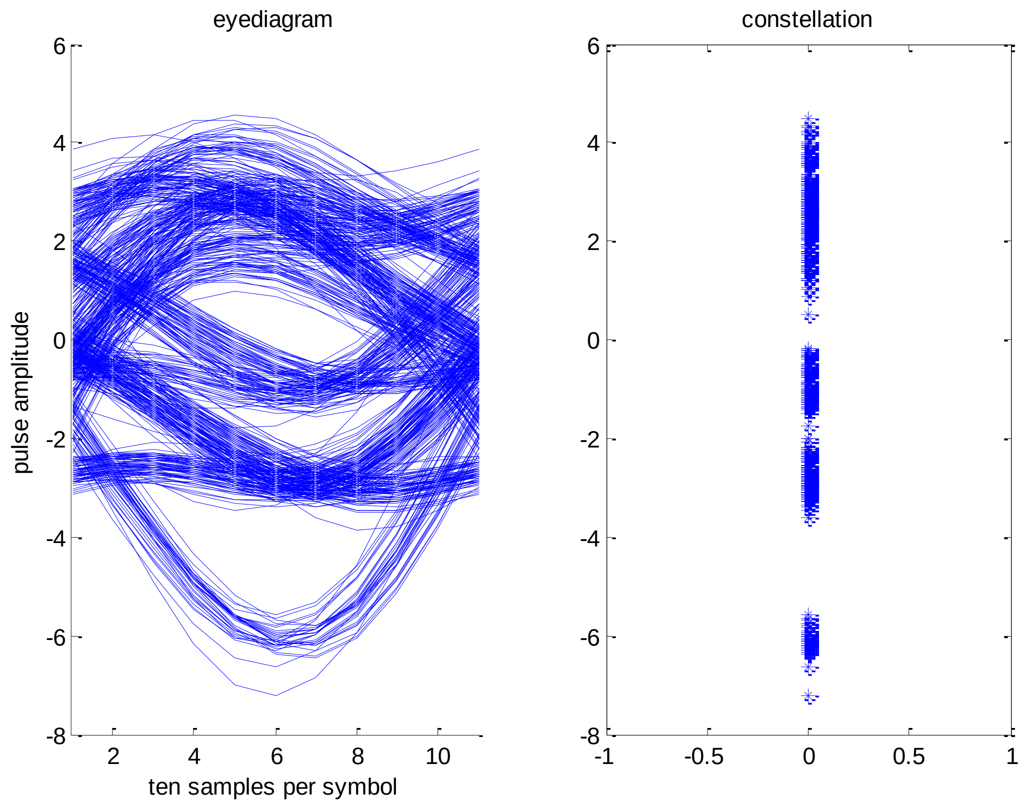

Fig. 6.16 eyediagram and constellation 3-PAM

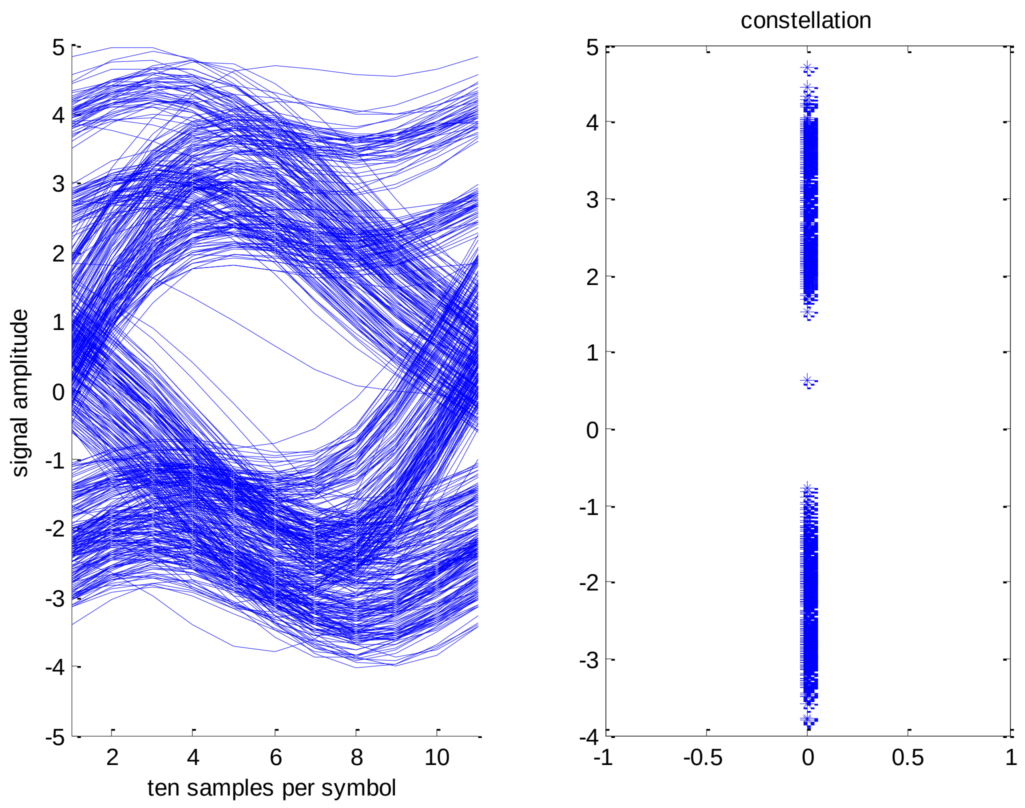

Fig. 6.17 eyediagram and constellation

6.5 Frame synchronism

As I said in chapter 5, the system has two ways for the frame synchronism one is sending a new level of our signal and the other is to send a Barker code. So we need two different solutions for each problem.

The lector will notice that from now on, I'm going to talk with the sequence that I send in the previous parts of this chapter it was not important because at that moment the unique important thing was to receive a Base band digital signal, what we put on this signal do not have much importance for filtering the signal and recover the synchronism that we can work only knowing that the signal is a BPSK or PAM signal.

6.5.1 Synchronism with a different signal level

In this case the system will receive ternary signal, two level are used for transmitting the signal and the lower level only says that this point is the beginning of the new frame.

For this the implementation is very easy, we have to suppose that our signal no has a big fading during the frame transmission and then I must find the minimum level of this signal in that frame. And that's all. The next figure explains this on the signal.

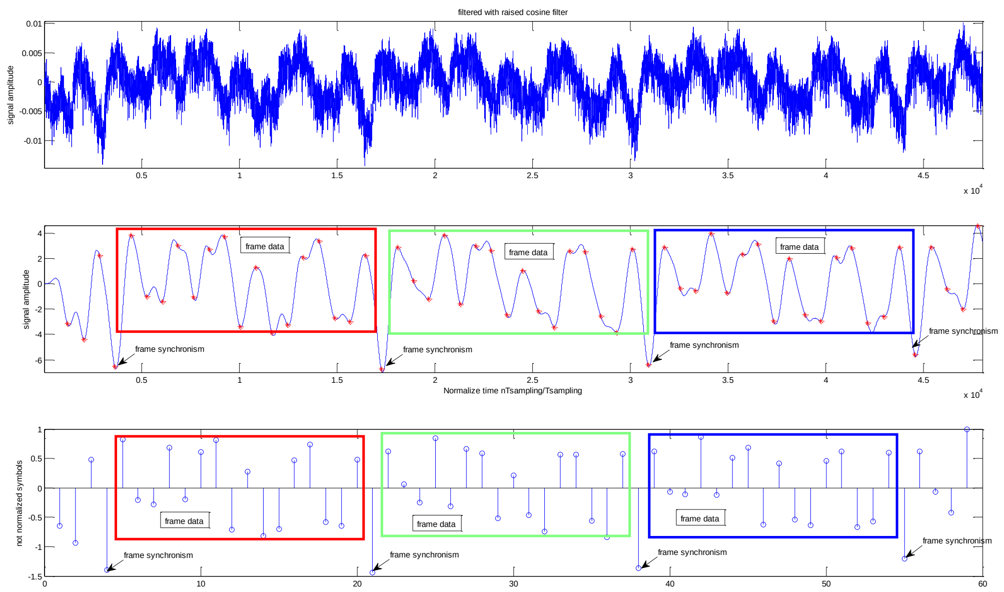

Fig. 6.18 Synchornizer detection

6.5.2 Synchronism using a Barker code

If we use a barker code for the frame synchornism, we have to proceded as the first case, i.e. the system must detect the barker code and then fix at the end of the code the beginning of the data frame.

To detect this code the easy way is correlate the received sequence with the original one, this has the problem that if we have bad luck the barker code can be repeated in the middle of the data sequence if that is the case I have to find which is the correct one, by compariing the two different sequences of data, maybe sometimes this will be imposible because we also have to find what data is sended to the token, but always is chosen the one with most power. This problem is not well solved at this moment

In the normal communication the system make bit stuffing, but in this case we can not do that due to the channel will lose the ortoghonality with the other channels.

To show this idea the reader mus look at the picture, in this figure the picture shows that we have one symbol sequence that has some barker codes inside, we correlate this signal with the barker code 1 -1 1 1 in this case and then the red lines show that they find this sequence.

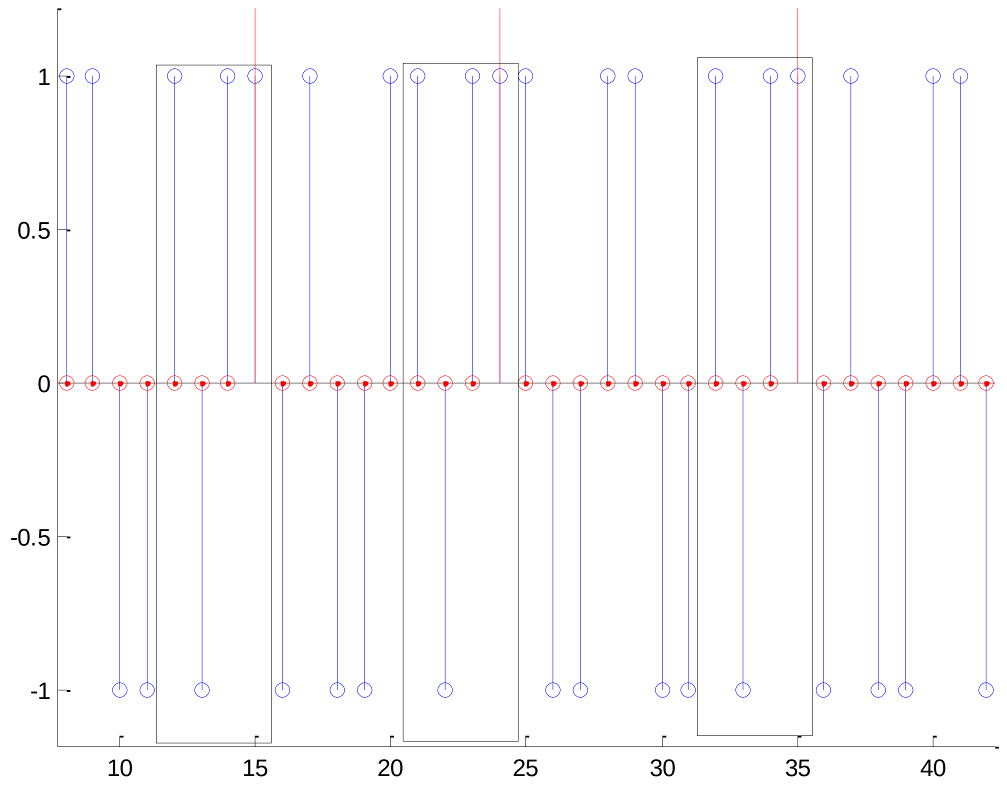

Fig. 6.19 detection of barker codes

The correlator is the same as used in the next section but in this case we do not know where the sequence begin so I must look at each symbol to know that. The barker code has a good response for that due to it's response to the autocorralation function is a delta, so until the code not is the same the values will be zero.

6.6 symbol detection using correlation

At this point of the modulator, I must discover which is the orthogonal code that the transmitter has send, this easy to make now, we perfectly know where the sequence begin so we do not have to look where the orthogonal code begin.

So the unique thing that we have to make is find the orthogonal code in which the system put in the signal, this is the same as despreading the signal without the code. This topic is well explained in [14], in this case we cannot work without knowing the beginning of the sequence, the explanation is simple we do not know when the transmissions begins, the token can be placed outside of the display, so when the token is placed on the screen it would be able to find the sequence. I write an example suppose that the token is in position 5 (0101 in binary) and another position is 1010 the sequence received without synchronization will be 0101010101010101 for the first case and 1010101010110101 0 in the second case then if we put the token in the screen and do not know when, suppose that we put the token in position 5 it can receive 101010101010 and decode it, and say I'm in ten but that's not true. So that is the reason, we need heathers for a correct demodulation.

Normally, in the literature you can find examples of despreading without the code without having synchronism. For us we make the same and another approximation, in our system we only use four orthogonal codes, so really we can correlate the signal with each of these codes.

With these bases we can construct our demodulator, as shown in the figure:

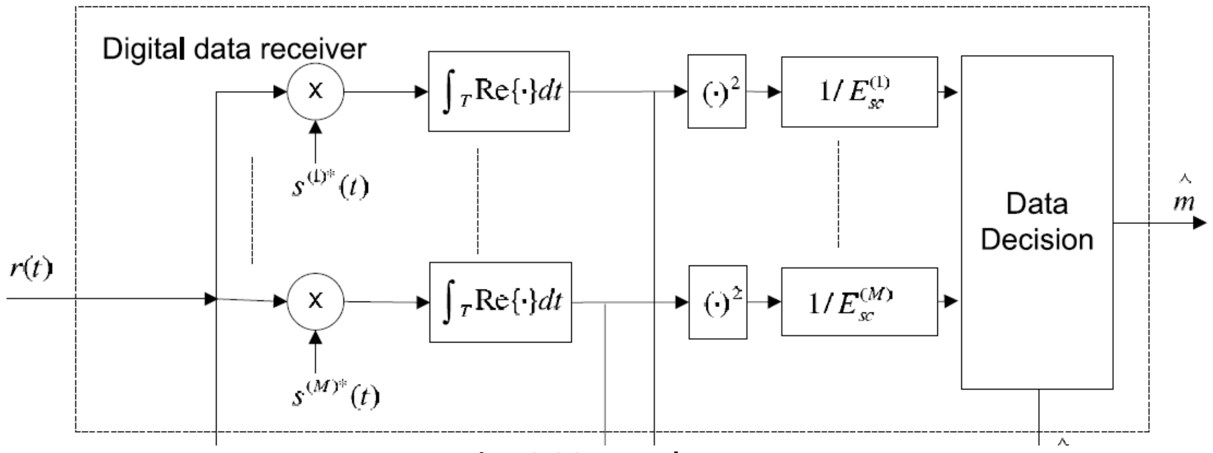

Fig. 6.20 correlator

In the system the symbols r(t) or the orthogonal sequences, are correlated by the orthogonal sequence that can be sent, then if the multiplied signals are not the same the result will be zero, in this case it's impossible because we have noise. Then we normalized each path with the correspondent power of his sequence, and finally there will be one that will have more power than the others. So that code is the code that we suppose that the transmitter has sent to us.

In the next figure is shown the output of the four path correlator, in this case the possible sequence is one of the four Walsh codes remember from the modulator chapter.

$$Wx = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{matrix} \quad (6.8)$$

With the idea of only show how work this correlator. I will divide the screen in for parts, then I will send to each division one different Walsh code, putting one synchronism symbol in each code then, this signal will be received for the other parts of the demodulator to finally arrive here. I start with the captured signal, but directly with symbols. I move the token in the middle the simulation, the reader can notice this movement in the picture of the received symbols

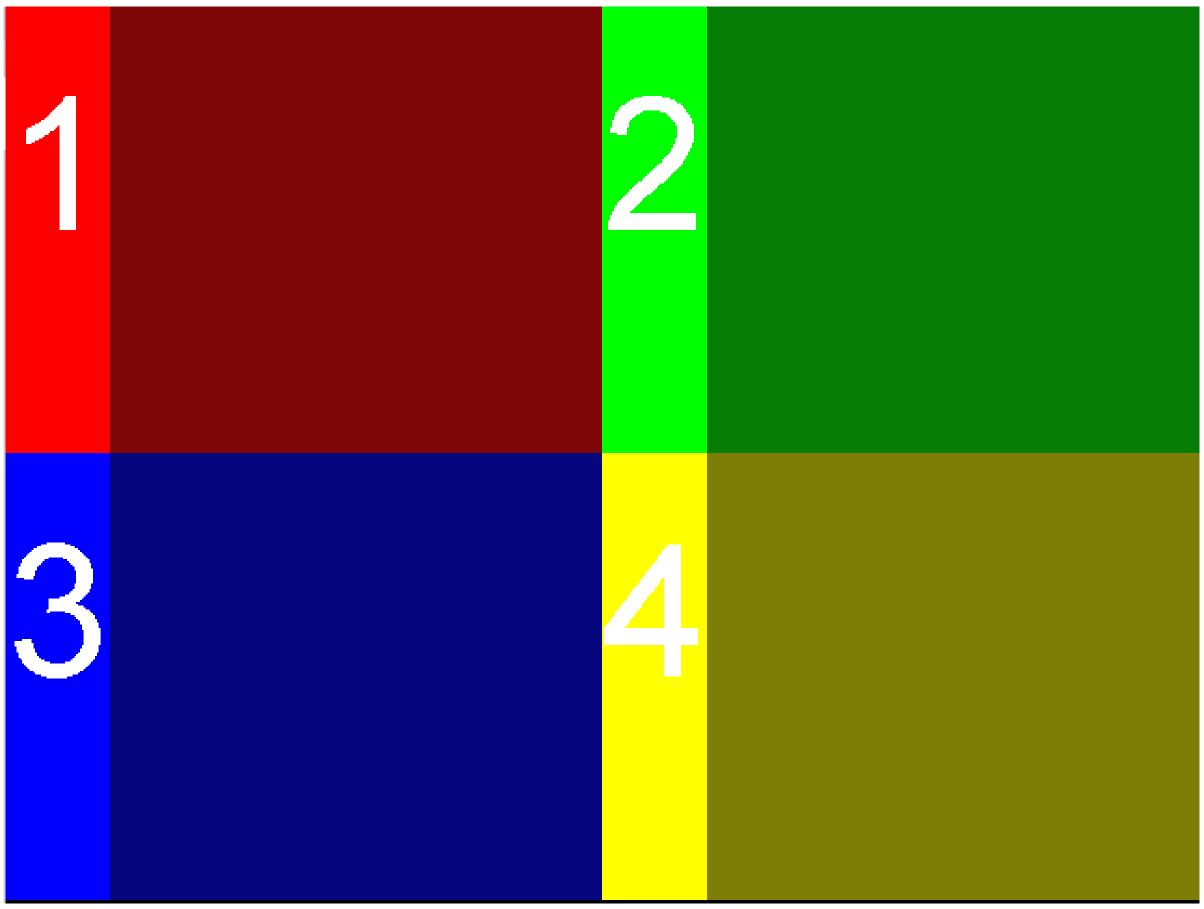

Fig. 6.21 divided screen

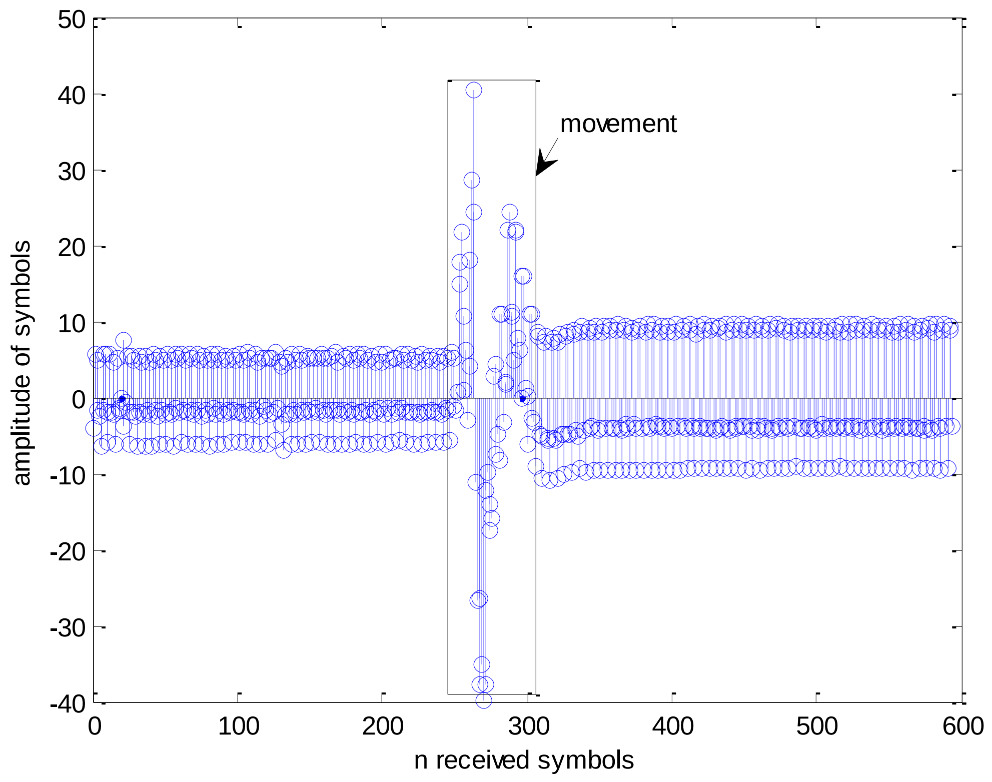

Fig. 6.22 Received symbols

The first symbols are placed in the green part of the screen, and the next position is the yellow part, the signal has the same level of colour in the pixels but the sensor receives more power in the yellow part, the second one than in the green, the first symbols.

Then I put these symbols into the correlator

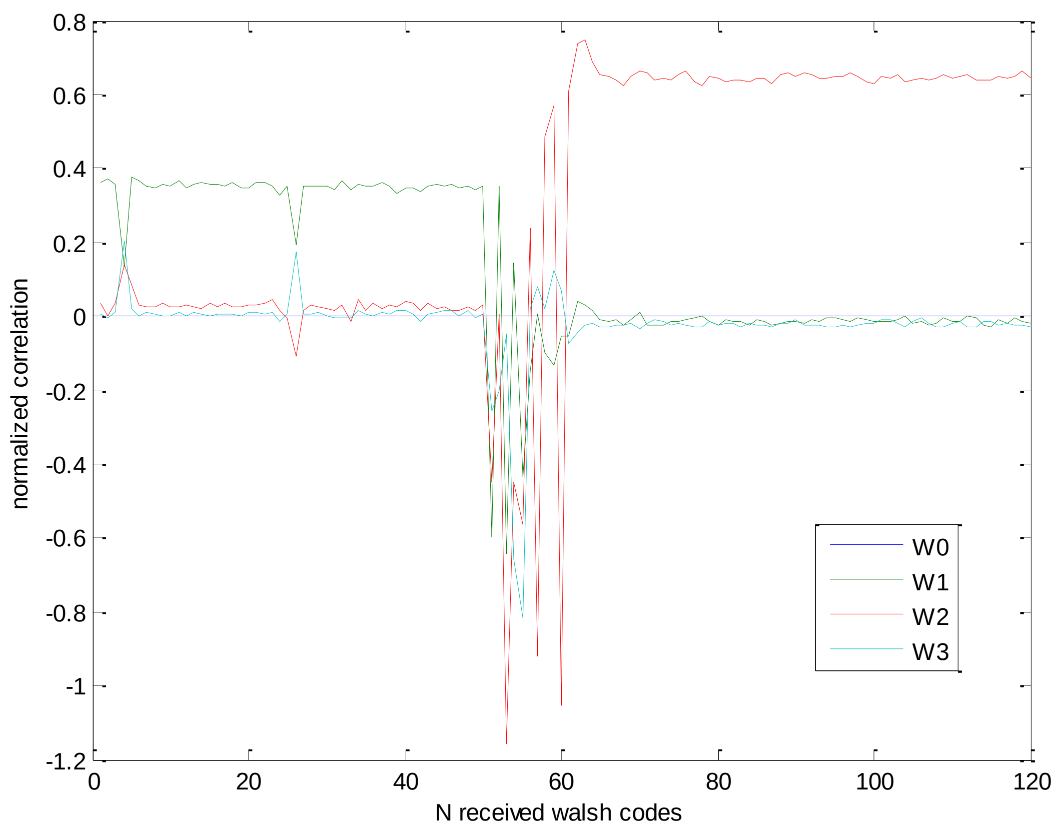

Fig. 6.23 correlator output

In the above figure you can see the output of the system's correlator. From this picture you can say that in the beginning the token was in the position that sends W1 then we can notice that in some time the token was not placed on the screen, and finally we can say without doubt that the token was placed on the part of the screen that sends the Walsh code 2.

For the next example we put the token in the middle of the regions that I move before the output of the correlator is that you can notice that our signal has contribution from this two signals so now, you can say that the token is in some place between this two regions.

For the ending with this topic, I want to say that another way to find the code that the display send is using Fast Fourier Transforms, Here is difficult to implement because the sequences are too short, but like it's explained in [15], it looks that would be nice to be implemented here.

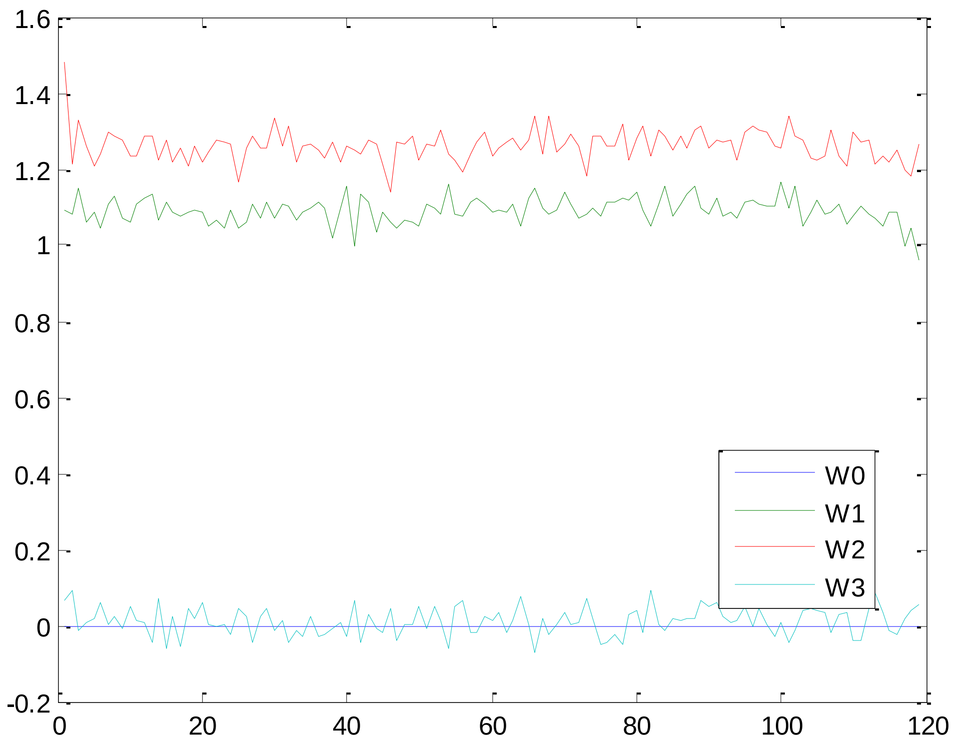

Fig. 6.24 token in the middle of two painted areas

6.7 The last demodulation

Finally, we need to recover the BPSK modulation that we apply to each Walsh code, but in this moment it’s very easy because we know the Walsh code that we have been used, so the only thing that must make here is dispreading the code.

When we have all the data, which is Walsh codes used, and the binary number that we put on the Walsh code. The system must call one decoder in order to the correct interpretation of this symbols, the implementation of this decoders is only to read in some memory the correspondence of what the system send with the signal that we receive.

This is exactly to make the opposite that we make in the modulator, there the system codify the positions into Walsh codes and BPSK modulation, so now we have to make the reverse way.

6.8 Symbol Equalization

If we look at the received constellation of the signal, we can see that this signal has a big excursion from their original values for as I think that this is not so important for having a good BER, because the correlator find the symbols quite good, but I think that will be good to use some Equalization in the channel.

In this case, I believe that the best option is to use decision directed adaptation, maybe at the beginning we can send some training sequence in order to find the first coefficients of the adaptive filter.

The problem that the system has is that the signal is to slow so for a good equalization we need to wait a lot of time if I want to collect enough symbols. But really I do not think that this will be a problem and the results will be quite gut.

The method used is easy to implement, only needs theory of adaptive filters. First of all we must make some estimation of the received signal then with this and the next input symbols we can make some equalizer that solve the problem of the bad received constellation. These methods are well described in [17] and [18] and there is a good Matlab example in this web [19].

With this topic I have had some discussion with my supervisor, and finally we decided not to implement. But here I only want to say, how should work a decision directed adaptator. Finally I try the example from this web to my modulator. In the next figure you can see the result of applying this.

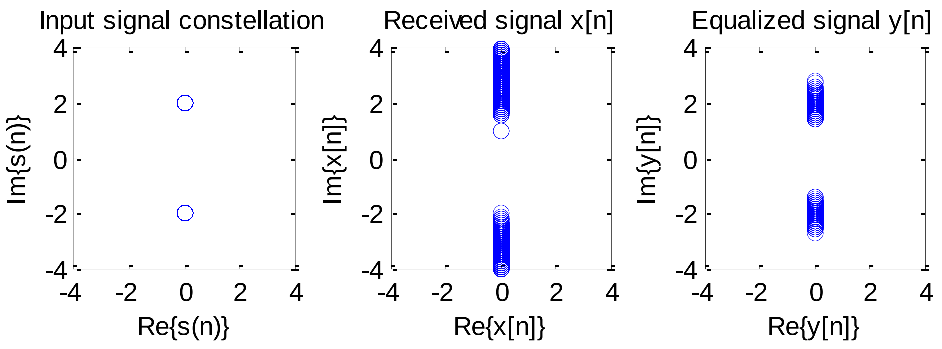

Fig. 6.25 constellation

So in this way is easy to reduce the variance of our received signal.

6.9 SNR Estimator

Signal to noise Ratio is required in all the communication channels, and this channel is not an exception. In this topic I have some problems with some estimators but finally we find two that go quite good, for this signal. In [20] you can find a good comparison of SNR estimation techniques.

Here I'm going to describe the two estimators that I used for the measurement of the SNR in my channel.

6.9.1 Maximum Likelihood SNR Estimator for digital Receiver.

This is well explained in [21] and the diagram of how it work is showed below

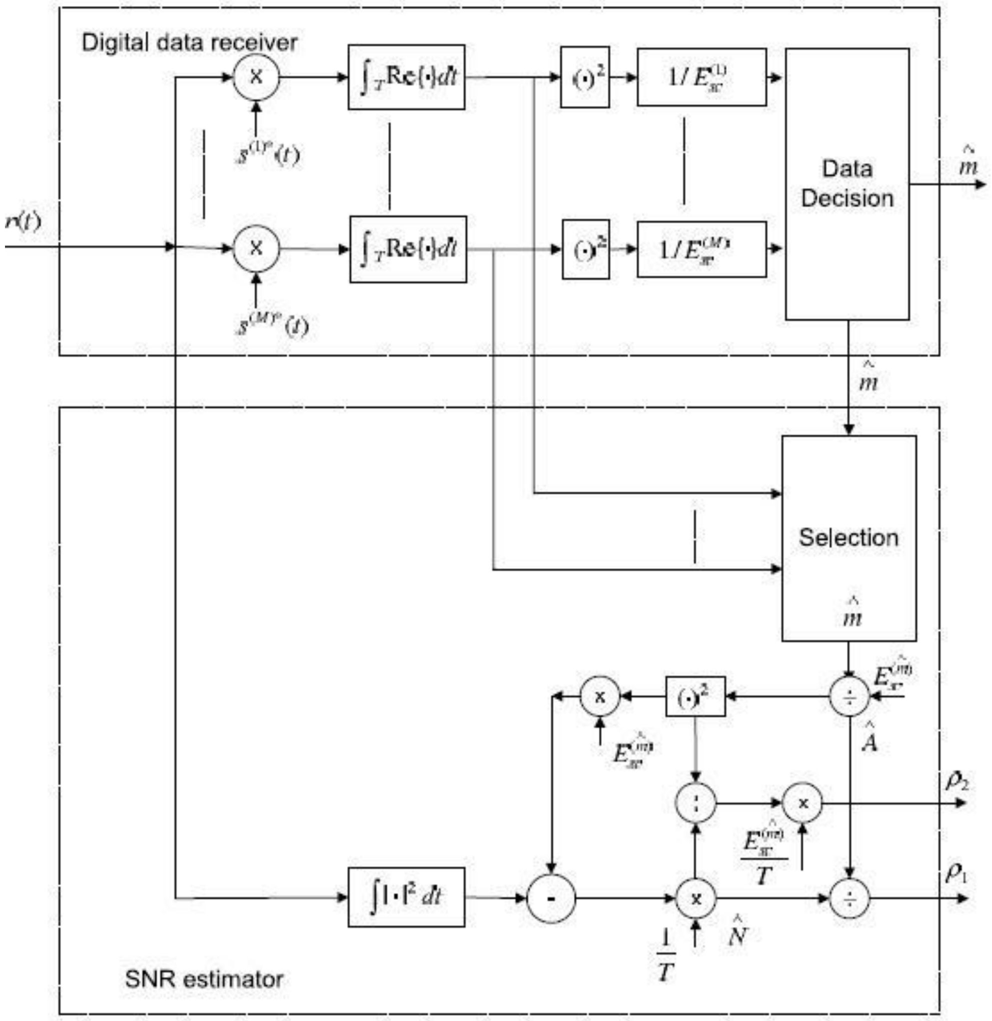

Fig. 6.26 SNR estimator.

First we need to find the estimation of the sequence of symbols that the receiver has received, in our case the Walsh codes, and then the next step has a lot of theory but really for us the only interesting thing is that we only need to apply one formula which is

$$SNR = \frac{K\sum_{k=1}^{K} r_k s_k^{(\dot{m})}}{E_{sd}^{(m)}\left(\sum_{k=1}^{K}\|r_k\|^2\right) - \left\|\sum_{k=1}^{K} r_k s_k^{(\dot{m})}\right\|^2} \quad (6.9)$$

The interpretation follows, K is the number of received symbols, r_k the received symbols, s_k is the estimated symbols and finally $E_{sd}^{(m)}$ is the energy of the estimated symbols, for us is 4 if we work only with Walsh codes, and that all for this part

6.9.2 A more practical one

In this case I normalized the received signal, then I estimated the possible transmitted symbol and I make Euclidean distance of this to sequences, finally invert this result and this is the second method that mainly use for the SNR estimation.

The results for the four colours are like this picture.

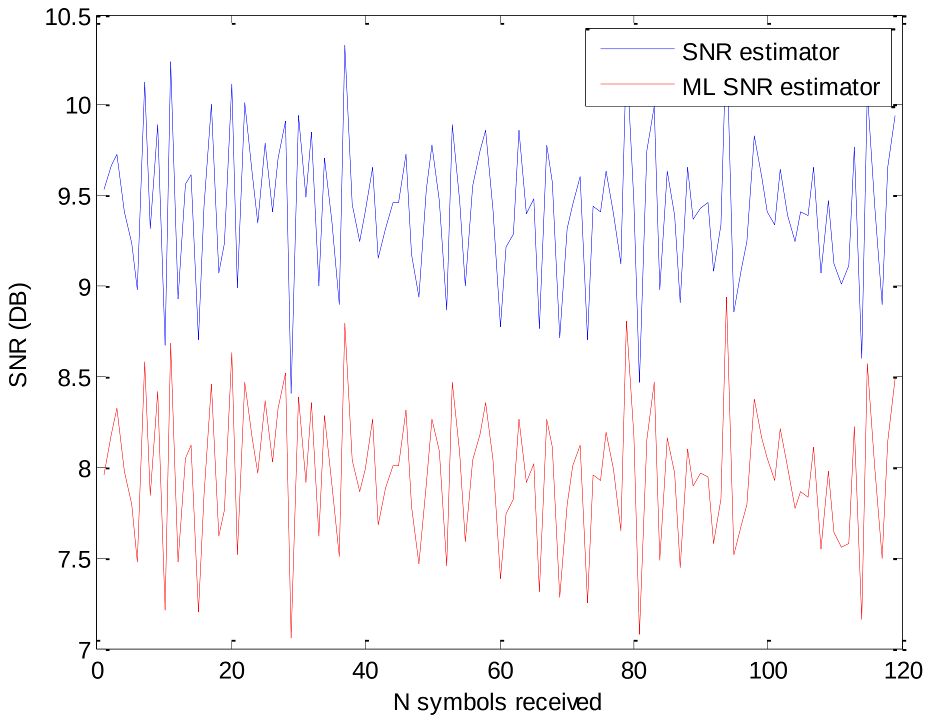

Fig. 6.27 SNR measure

In the above picture, you can see that we have a good SNR, but in this case we have to consider, that the ML is more conservator that the other, and finally this good conditions of SNR change when the area is small, and the token receive contributions from other areas.

But you can see that this to plots, they having different values but the same shape, so it's the important thing for the estimators

The difference between this to is that the first is more realistic and is faster for the recognition on added received signals.

6.10 BER estimator

There are some receivers that have some BER estimator implemented, but in this case I think that this will be better to make this measurements on the easy way that sending one signal and then look what I receive and compare with the original transmitted signal.

Chapter 7

Results and channel characteristics

In this chapter, the reader can see the results of all the system working together. I.e. we divided the LCD and then send one signal to each position, and the token which is placed in one region, it must capture that signal and demodulate it, so the parameters that we are going to show are the channel capacity, the SNR and the BER, and finally one test for the invisibility of the signal for the final user of the system, this is the player.

7.1 Channel capacity

If I'm talking for a normal channel, it's easy to see how to calculate the capacity of the channel. We send symbols every time that the LCD redraws so if we only send binary symbols for a base and system, the result it's 60 bps, because the screen changes at the rate of 60 Hz, so please notice here that if we have screens with faster rate of refresh this will be higher.

But in this case the better that we can do to test this is answer this question, how many time we need to find the position? To answer this question, I must know how many symbols we need to code one position, and then I also know how many time I need for send one symbol, this is the inverse of the refresh rate, 16.7 ms in this case. To make that, first remember the form of one frame

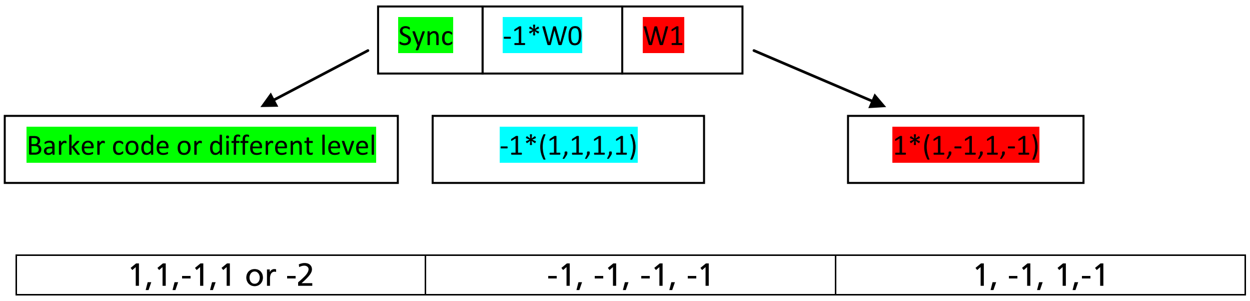

Fig. 7.1 example of frames

The frames consists in one part of synchronism, (one or 4 symbols), and then a succession of n orthogonal codes, that each of them use 4 symbols. This n depends of how many positions we need to code as said equations 7.1 and 7.2. That has been deduced in chapter 5

$$N = 4 * 2^n \ (7.1)$$

$$N = 4^n * 2^n \ (7.2)$$

So if we need 64 positions, using the same Walsh code in each frame, this is equation 7.1 I need to put n=4, and for the use of different Walsh codes in each frame I need 2 Walsh codes per frame. So the duration of one frame will be

$$T = duration\ of\ synchronism + duration\ of\ information\ (7.3)$$

The last equation describes how much time I need for send one position. The duration of synchronism is the n symbols that we use for synchronism divided by the refresh rate of the LCD Fr, In this case is 60Hz.

$$duration\ of\ synchronism = \frac{n\_symbols}{F_r}\ (7.4)$$

And for the duration of information is if I use n orthogonal codes of length 4 and I can use the four codes

$$duration\ of\ information = \ n * \frac{4}{F_r}\ (7.5)$$

So for example, we are playing chess, so I need 64 positions, the number of Walsh codes that I need if I can use different codes is 2 and the number that I need using the same code is 4.

So finally the duration of information is 2*4/60=133.3 ms in the first case and 4*4/60=266.6 ms for the second place. Finally at this time I must add the time for sequence of synchronism that it can be 1/60 or 4/60 seconds.

So the minimum time is if I use the optimum code of different Walsh codes in a frame and one symbol for synchronism. This is 150 ms and for the worst case using 4 codes and a long sequence for synchronization this is 333.3 ms. so in an ideal case we can calculate 6.6 positions per second.

Due to the low pass filter that has the audio device, I cannot use the four Walsh codes, so it limit me to only use three so I need to put words on each frame to code the 64 positions or to use another orthogonal codes.

To increase this we can think in smarter codification of the position. But one thing that really decreases this time will be use LCD that has faster refresh rate. Because in that case the duration of the symbol will decrease and the sequence will be faster to transmit.

The reader can also think that one thing is to use more orthogonal sequences, but since we apply the BPSK modulation to the orthogonal sequence this doesn't solve the problem. The answer is easy, for having k orthogonal sequences, the length of this sequences must be k, so for example if have Walsh codes, 4 orthogonal sequences have 4 symbols, 8 have 8 symbols, 16 have 16 symbols and so on. So let's make the operation. We need to codify 100 boxes in the screen, so if we use 4 orthogonal codes we need to use 3 of them, if we use five we only need two of them, if use six also two, for 8 also two. But then if we translate it to bits in the first case I need 12 bits, in the second 10, in the third 12 and for the last I need 16, so really the gain it's only in one case. Now the reader can see this, in the next figure, I'm going to show positions that I can code versus codes I need, and then bits I need per number of positions.

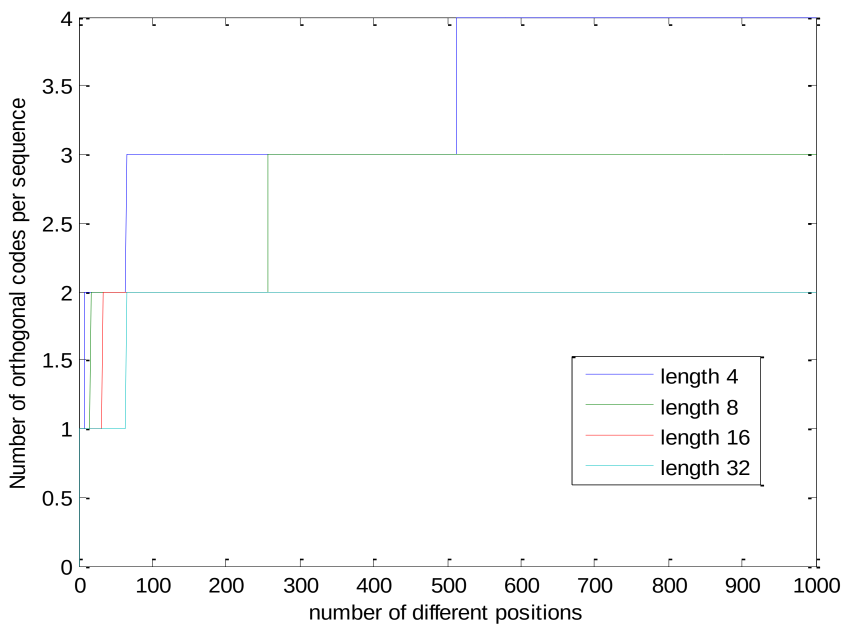

Fig. 7.2 computation of 7.2

In the above figure we use 7.1 to calculate how many Walsh codes I need, giving the positions that I divide the screen, as logical the one that needs more is the Walsh code of length four because it only has 4 codes, but if translate this to the bits that we need to send the sequence, this is the one that need less bits.

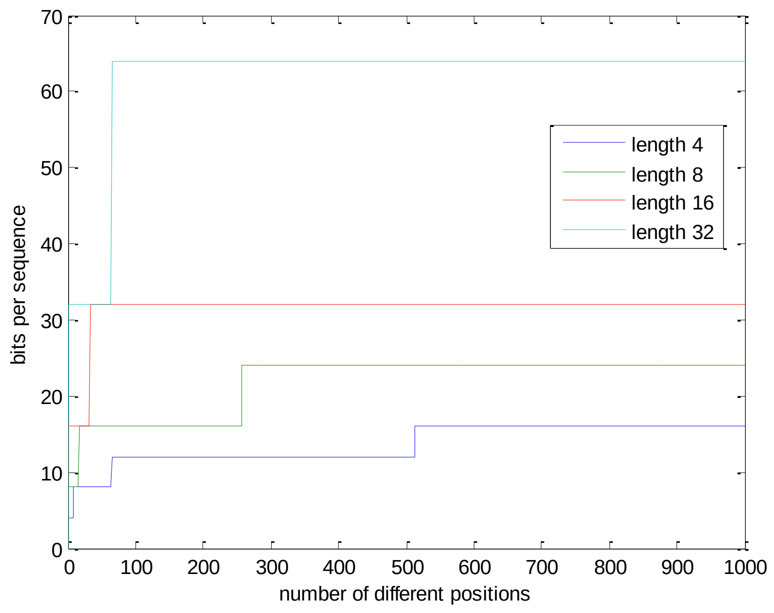

Fig. 7.2 translate to used bits

The same thing we have for 7.1

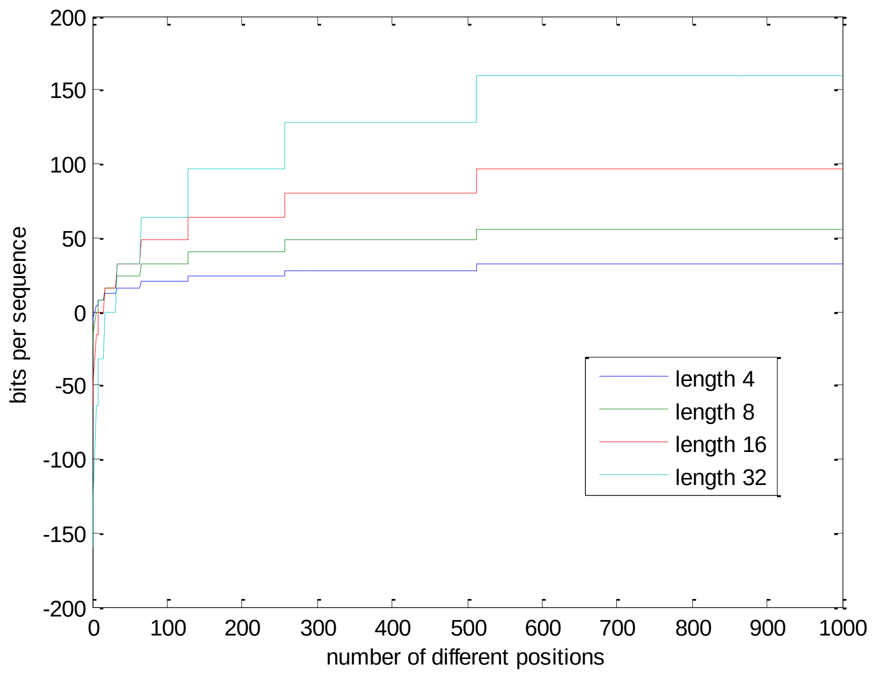

Fig. 7.3 translate to used bits 7.1

So really we must use the shorter Walsh codes for saving bits. It's like the BPSK term 2^n makes some compensation, because is sure that with only four Walsh codes. We can form fewer combinations that with 8, but also with 8 we can form less combinations of this term that with 4. So this is the explanation, will be a good exercise to demonstrate it mathematically. In 7.3 we can see that. This is only for Walsh code of length2^k.

$$number\ of\ bits\ = ceil\left(\frac{log2(N\_positions)}{k+1}\right) * 2^k\ (7.3)$$

So the best way for improving this, is modify the refresh rate to one higher.

In this moment the system that works without significant errors is one that uses three orthogonal sequences of length four. The type of frame is repeating the Walsh code and I divide the screen in 48 cells. I have to remove one orthogonal code because it was continuous and the audio device filters that code. I use a different level for synchronism so the time that I need to send the full sequence is

$$duration\ of\ synchronism = \frac{n\_symbols}{F_r} = 16.7\ ms$$

$$duration\ of\ information = \ 4 * \frac{4}{F_r} = 266.7\ ms$$

$$T = duration\ of\ synchronism + duration\ of\ information = 283.7\ ms$$

Finally, in the Matlab implementation of this receiver, I need the double and two symbols more time, this is (17+17+2)/60, so I need near half a second. That is due that I do not know when the frame begins so in other to be sure that I can get a complete frame I must wait all this time looking at the channel. But in real time implantation it's not necessary if the token is always looking at the channel. If not, the thing that we can make is waiting only the time that the displays need for sending all the frame and add two symbols more. Then we know the begin of one frame , and we also know that the cell always is sending the same frame so the only thing that I must make is some permutation of this cached symbols to get the frame in the correct order.

With these results, I want to say that at this moment, this is good for localization games that are quite static, like strategy games (i.e. Risk, chess). But if the system needs to be implemented in fast skill games, for example put the token in that place then put another here, then the other…, it has to be implemented at the end of the token movement, and then check if the position is correct. This is also because the token needs that during this time the signal that it received was stable. If not, it has errors and fading of the signal.

7.2 SNR measures

I made only this measures for some boxes in the cell, I put the token in two different positions, in the middle of the cell and then in the boundary of that cell, where I have interference signals.

I put the token in the cell of column 5 and row 3, the estimated and received signal is shown below

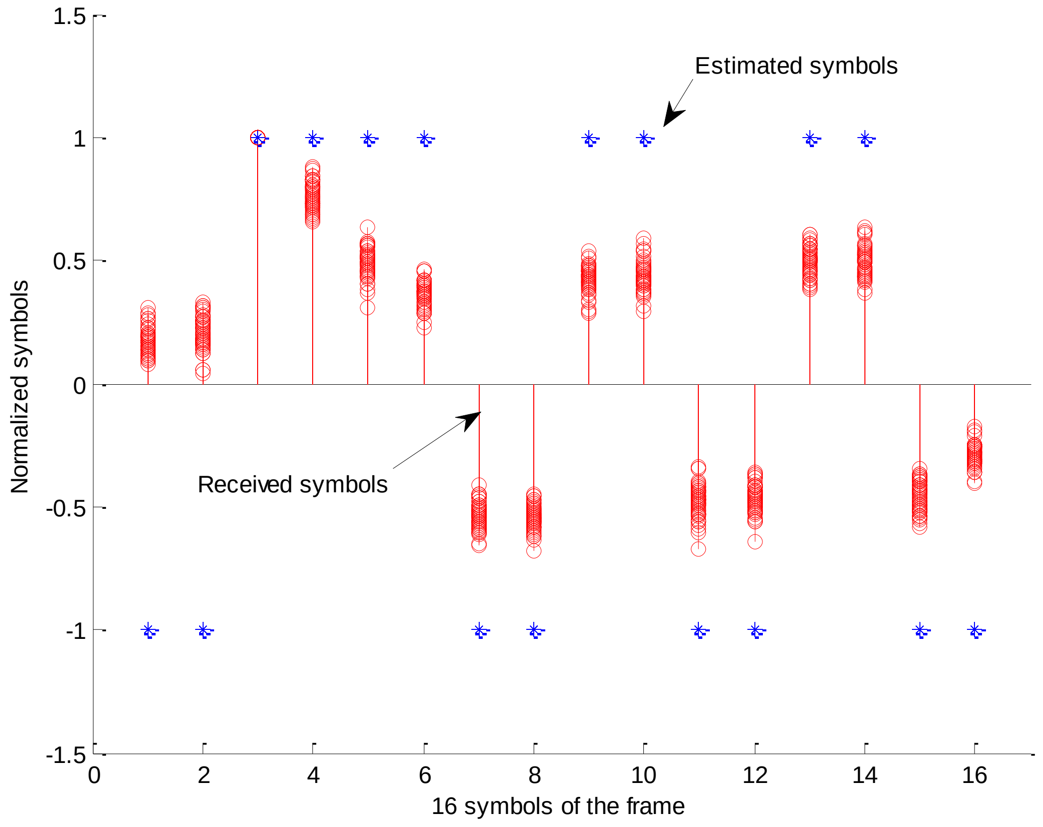

Fig. 7.4 Received and estimated signal

You can see the variation of the received signal, and inclusive two errors in the symbol, but thanks of the detector by correlation and of using the same Walsh code the system can correct that

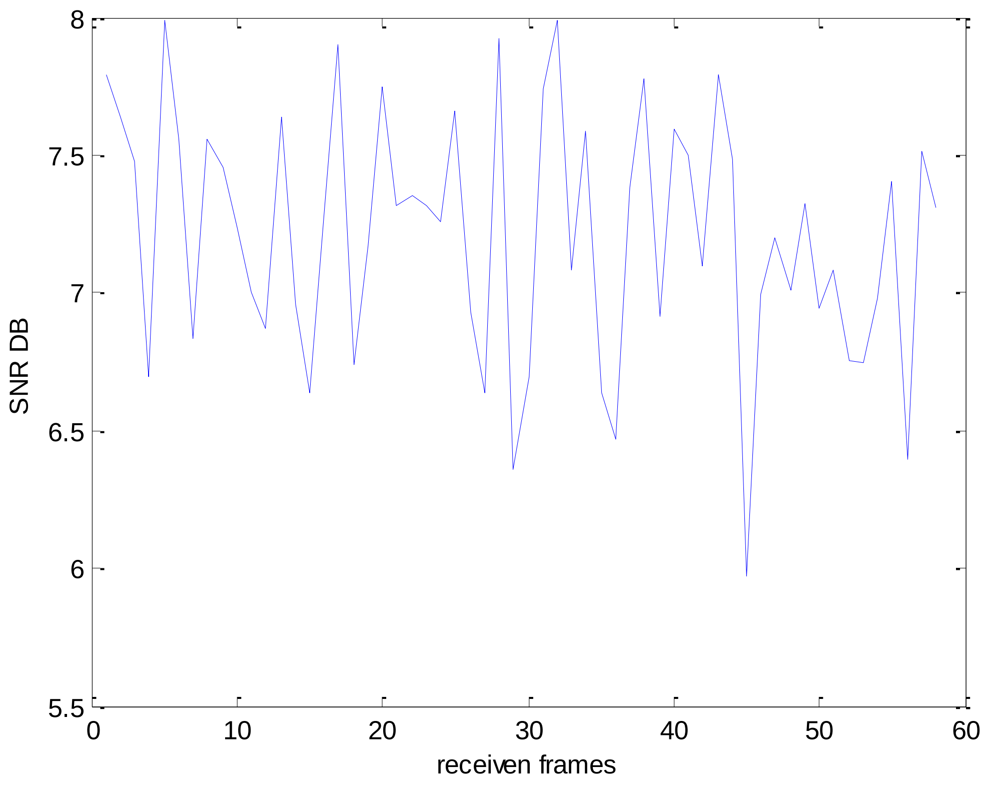

Fig. 7.5 SNR measure

We apply Maximum likelihood estimator, really this method don't apply any normalization of the symbol, but my supervisor consider to apply always at the entry of the system, so I divide always by the maximum symbol, finally the SNR has a lot of variance. The problem is due to synchronization using another level, so will be better to implant it with Walsh codes.

At this point our group think that the problem is due to the audio device, because in the oscilloscope we can see it better, so the problem is that the audio device filters the low frequencies.

Then I'm going to show the SNR when I receive two signals, not in the middle of the hexagon like before. I put the cell between the cells 3 and 4 of the row 4, The system detect the (4,3). The estimated and received frames are shown. In the next figures.

Really in this case it's difficult to decide which symbol the system send, but the correlator detects it. Now the system has decreased the SNR because it has two signals so one is only interference. for the system

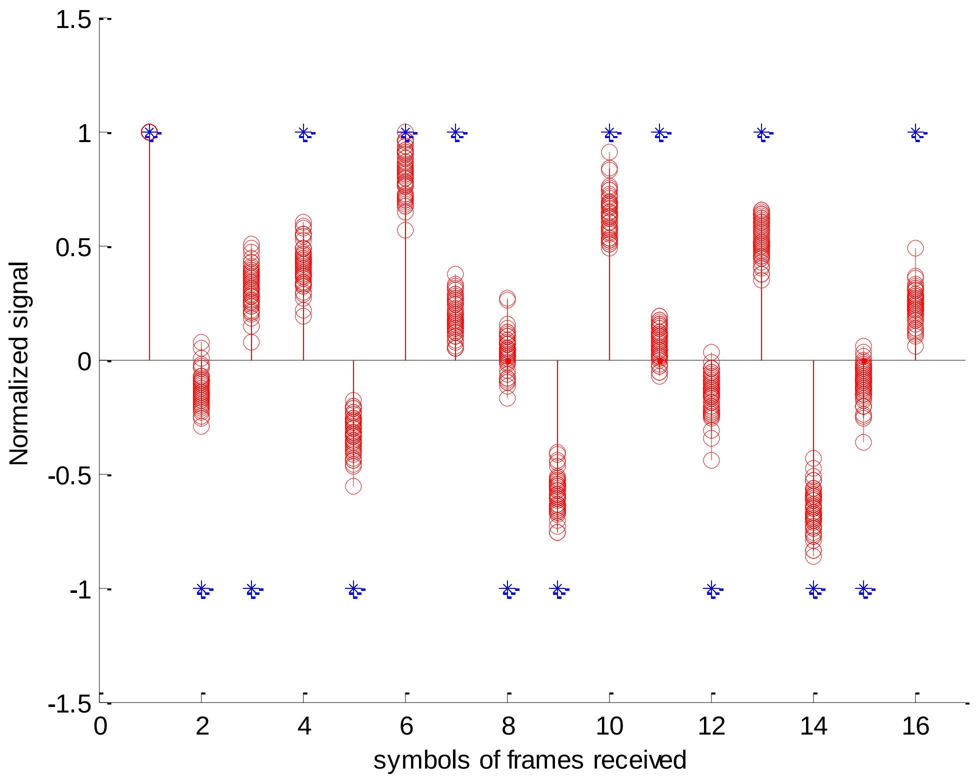

Fig. 7.6 Received and detected signal

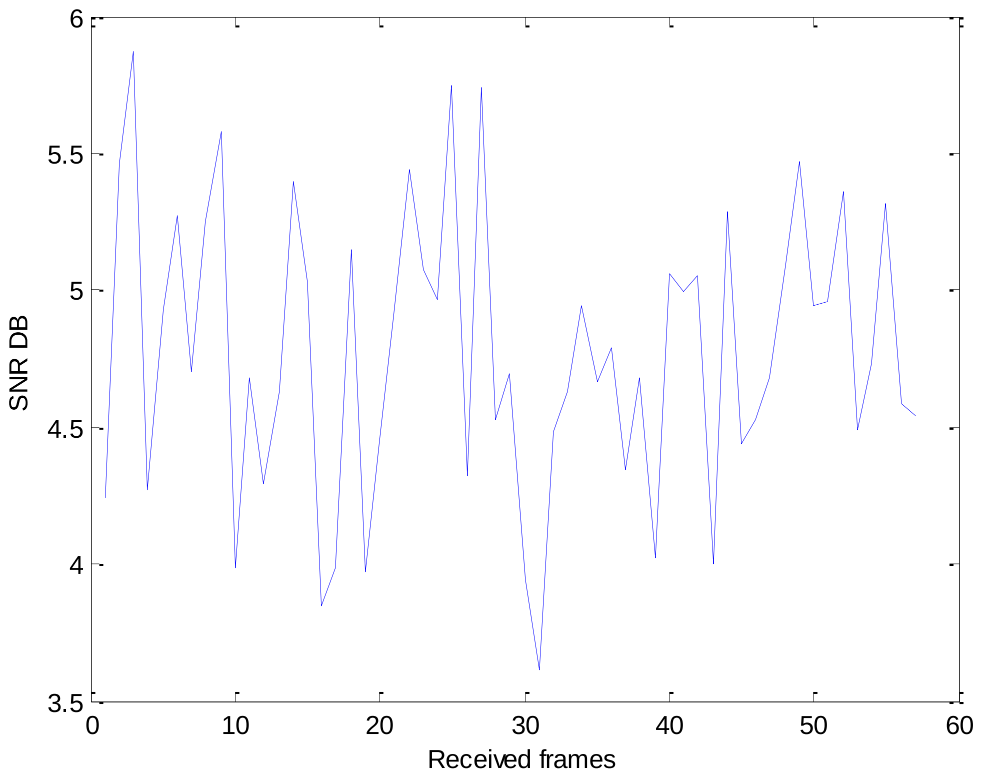

Fig. 7.7 SNR

7.3 BER measures

To make a good SNR measurement will be necessary, because I do not have enough time for making and accurate BER analysis of this system.

The measures that I made are quite good for this system. I made two types the first is that has best results because it's when the token is static and catching the received signal all the time, this has not errors and for some cells the BER is lower than 0.01 %, there are cells that this is increased until 1-1.5 % of errors. In this case this is very rare but exists so it will be remove in future implementations.

The other is the sensor only looks the channel the time for catching two frames and this has 1 % of BER.

For making this measurement I put the sensor in one area that I know the signal, and compute the errors that I have when the signal is received and divide it by the number total of measures that I make and that is the BER.

So will be interesting to make some BER analysis for different images and codes

7.4 Detection of the movement

Like I said before, the token must be static during the time that it is looking at the LCD and waiting for the full sequence. Really this is not true if the area which one the token is moving, if has the same the brightness the token can detect it, but when the area has a lot of brightness change for example has one black colour opposite to one black colour when the token move from the white to the black this is a big fading of the signal, so the token will lose all. So it's important to adapt this changes quickly, really is difficult because the change is instantly an can only affect one bit. The next will show you one image that can happen this.

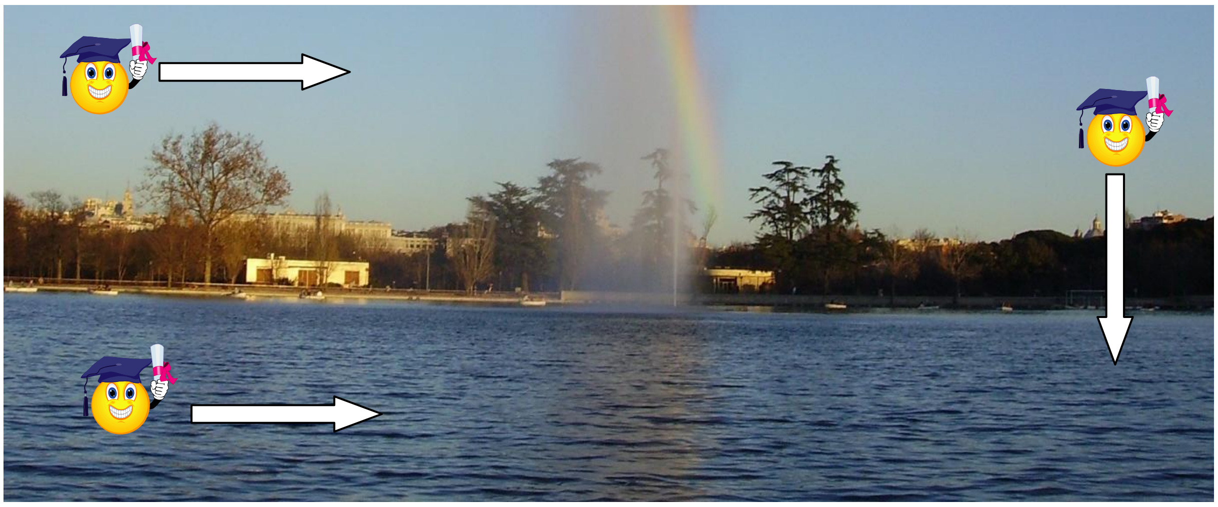
Fig. 7.8 problems with movement

In the above figure you can three tokens and the direction of their movement, the ones in the left move to the right and his areas of movement has the same level, so in this case if the token move slowly, and the sequence is not so much long, it can detect it, but for the token on the left the signal will have a fading from moving the one side to the other, because it goes crossing one black area and the signal falls inside so it cannot detect this situation in the next figure you can see the fading of the signal. Please notice that if the token is static on the black area it can detect the signal.

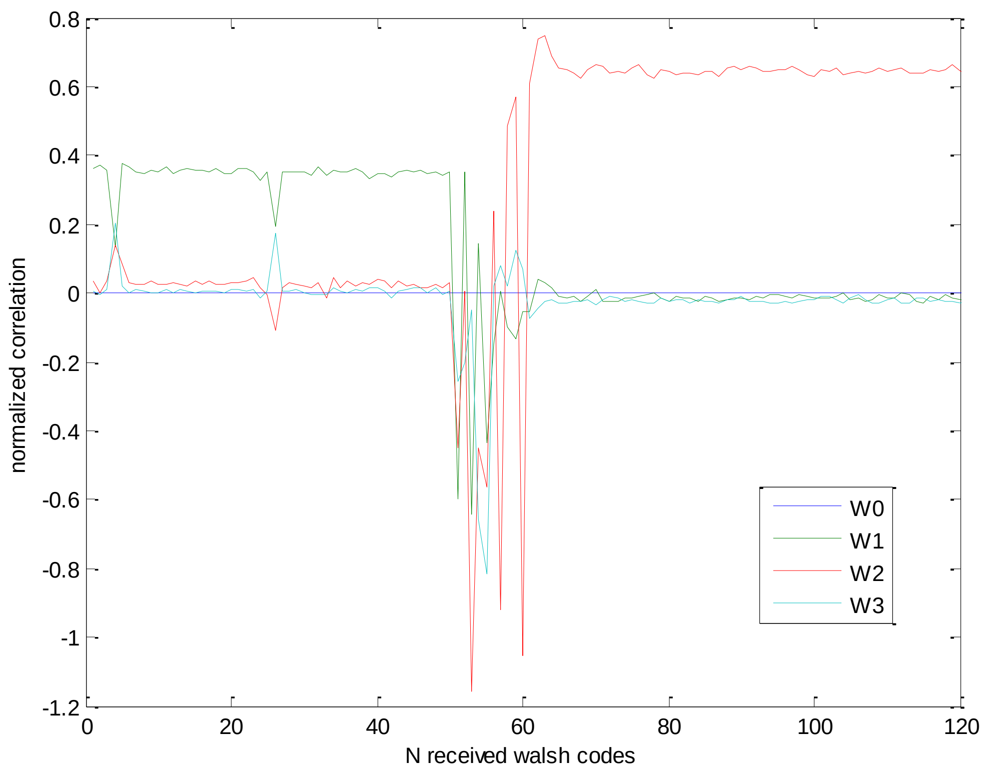

Fig. 7.9 fading signal

7.5 Invisibility of the signal

I made an small test to my partners in my department, For the experiment I use this image on the LCD, I divide it on 64, and 48 cells, one using all the Orthogonal codes, and then the other avoiding the continuous code, I also use different types of synchronism barker codes and the different level, And I display the image like in the next figure, and every cell send one code. Finally I tried different RGB values of brightness. I.e. I send the same sequence but first to code and one if the level was 129 I send 130, and then I tried with 132, so the difference between levels of brightness was first only one level, then two and so on until four. I have only 256 possibilities in each colour.

The results were quite fine, if I only use the first two levels of separation between the symbol 1 and -1, only few people can see this, but in a normal game that they do not

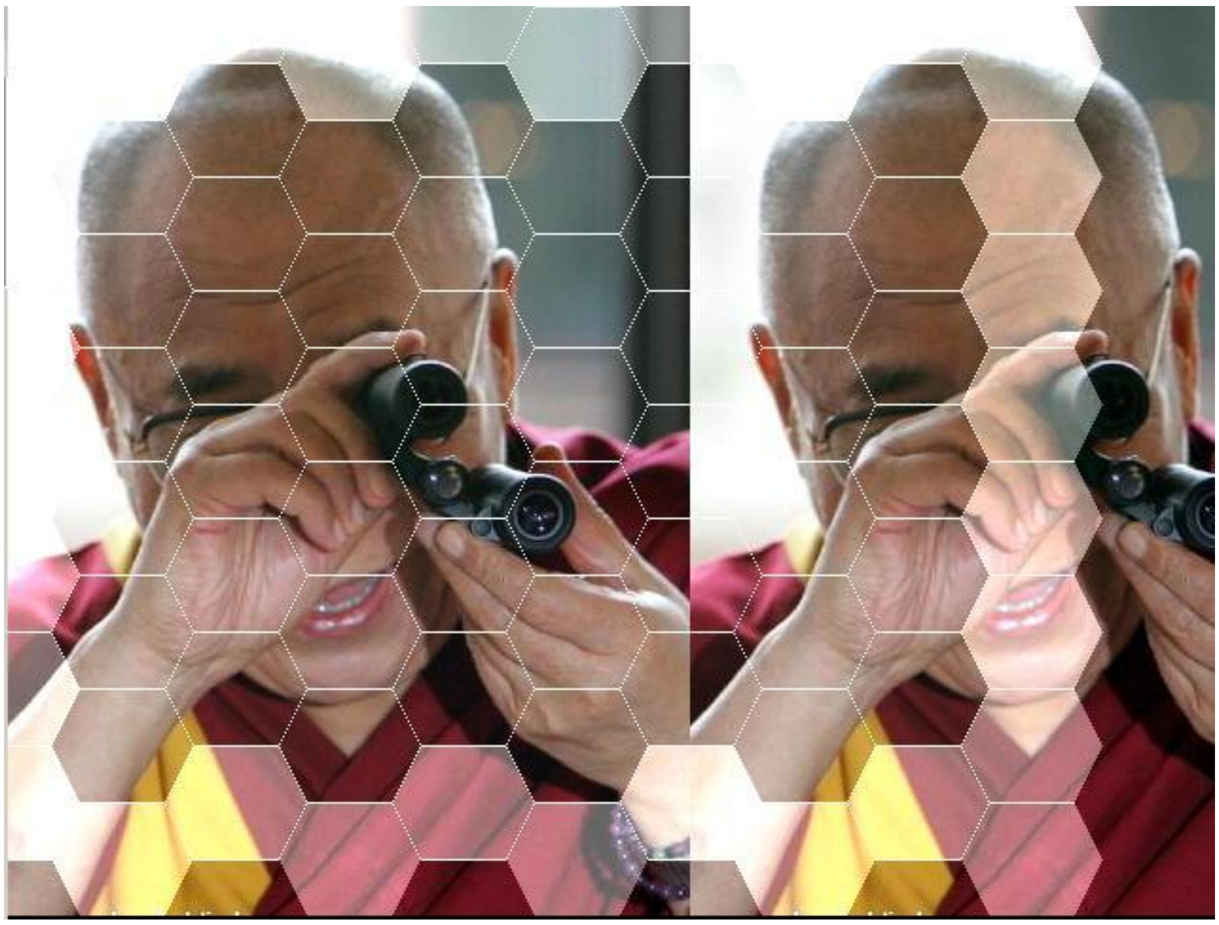

Fig. 7.10 LCD display

Look so nearly to the screen the people do not detect the sequence, if the separation of levels was more than three the signal was quite visible and sometimes annoying for the player.

Finally I can notice that the people can detect easy the flicker if the sequence was shorter than the others cases. Another thing that people can detect quite easy was if the sequence has a very low frequency, these sequences were easy to detect than the sequences with more changes on it.

In order to finish this work, I think that will be good to make a test for invisibility of the signal with more people, I have not enough time for making this.

The conclusions of this test are, the sequence must have a lot of changes on it, so I must remove the continuous codes and the other and most important is that for a good playing the difference with brightness levels form this scale of 256 levels must be lower than three. So I only use two or 1 level of separation between the values +1 and -1 of the transmitted sequence.

Finally if the area on the image was on a clear area the user can see more the flicker in the image that produces the sequence.

Chapter 8

Future work

In the next lines I'm going to describe how to improve some weaknesses of the system.

8.1 Add channels to the system

The main problem here is the time that the token need to detect the position because; it has to wait until the entire frame is transmitted. The first thing that I can make is looking for LCD that has faster refresh rate than 60 Hz, Now in the market is easy to find between 85 HZ and 100 Hz. So this is a good way for improve it trying to find monitors with this high rates.

Another solution is this, we code the information of each division, by decreasing or incrementing the brightness of the pixels that are in each division, but each pixel has the colours RGB system, red, green and blue. So we can, send the information using the colours.

Example, Now for coding +1 if the pixel has (R, G, B) (123,114,100), I send (124,115,100) and for -1 the same (122, 113, 99), so really I can use two channels more, for example, now I'm going to code the +1 an -1 at the same time (124,113,100), and only one time I code two symbols and I also have the blue component for code another symbol.

But to make this I need one thing that I will describe. Now, I cannot send the information in each colour, because now for separate that I need three different Walsh codes and then 3 more for the neighbour cell and finally, and using hexagonal structure for LCD division, another for the other neighbour, so I need 9 orthogonal codes for make that, at minimum. Or smarter distributions of the codes that I have made.

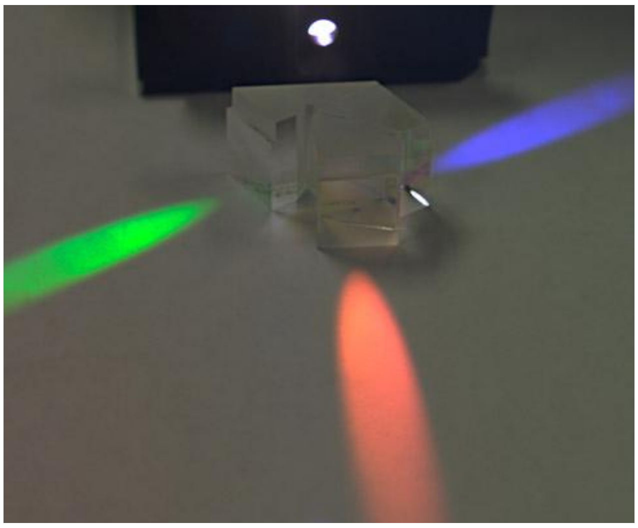

Fig. 8.1 Prism

Really, it can be easy and with less orthogonal codes to make that, but the problem is to be sure that I can separate the channels. I think that the best way is to separate this colours that the screen draw. To separate this I think that will be good to use a colour filter array (CFA), colour filter mosaic or a 3-CCD Dichroic prism, this systems are the ones that the video cameras are using now for separate the RGB components of the captured image/light. With this I can send different information in each colour. Because the prism separates this three colours. So I do not need to put one orthogonal code per colour in order to separate this.

If I make this instead of one channel the system will have three so in theory, it will go three times faster.

I think that the separation of colours will work well, because when the LCD paint the image, this image is painted combining three colours red, green and blue, so I believe that it's easy to separate this again.

I try to separate two colour channels using orthogonal codes, but to obtain acceptable results the codes must be longer that the ones I use. And for short codes the only sequence that I can detect was the sequence that was modulated in the colour that has more power (higher value on RGB) in the area that I put the token.

I also want to say that, maybe, the three colours can be separated using techniques of time multiplex, because each colour is turning on at a different time, but this is only an hypothesis.

8.2 Movement detection

The system can find movements of the token but really cannot measure the velocity of one token; the reasons are that when the token move from one darkness area to another more brightness the signal has changes or fading. These changes are a quite fast a difficult to solve because they can only occur in one bit of the hall sequence. In Order to avoid this sequence has to be shorter.

But I would recommend not dividing the screen not using a regular hexagonal grid. And use segmentation of the image. And then make the joint of groups by the neighbours with the same level of brightness. So in this case if the token is moving inside one area the signal that he will receive is the same in fact of power, but when it moves to another area that it's not the same region, this will be an area with more or low brightness so the token can detect this and can say, now the token is movement, for catching the area the token has to be inside the other or some time static in this new conditions of brightness levels. Finally I show in the next picture the difference between this to divisions. I have not enough time for trying this.

This I think that is the best way for solving this problem, and the also with this we know the areas that has more brightness in our picture, so this will be good in order to make more invisible the signal, adding more or less levels depending on the area.

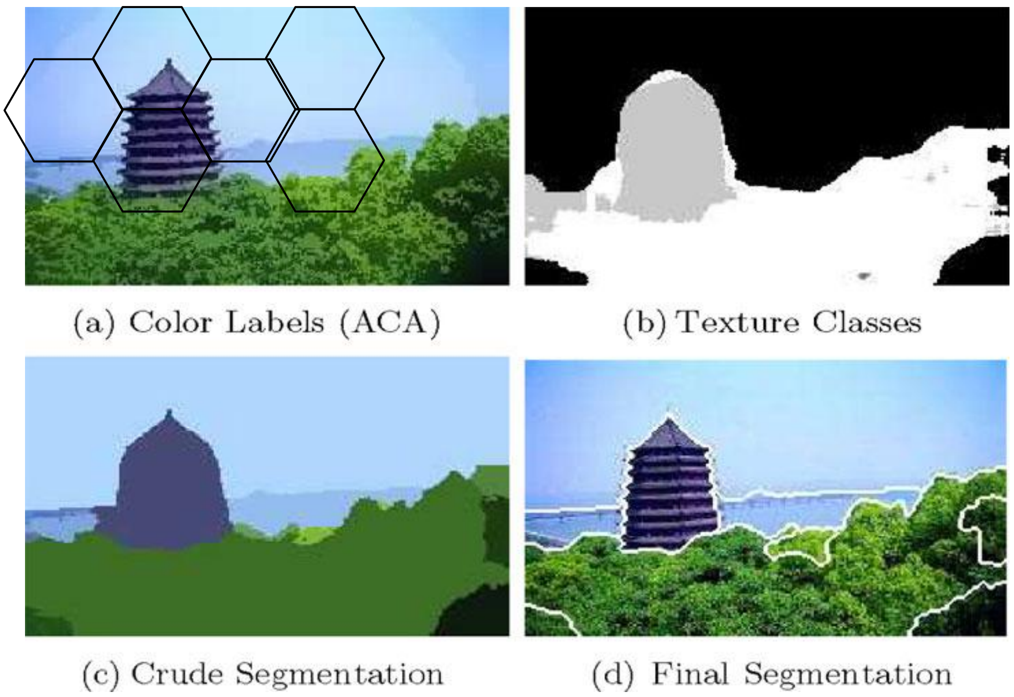

Fig. 8.2 Segmentation of image

8.3 Test for Health

It Is important to make another test of the invisibility of the signal to more people that I make here in Fraunhofer, because now the signal is quite invisible for the player, but the problem can be more serious. The signal in this system is produced by setting different values of brightness in the different regions that the screen is divided. So this creates some flicker light. At the frequencies the system is working from 0 to 60 Hz it can produce reflex epilepsy.

This type of epilepsy []is Generated by flashing lights, alternating light/dark lines or synchronous flickers, or is triggered at times by reading, eating or immersion in hot water. So this how the system embedded the signal in the LCD.

So I recommend that for the final implementation of this system. A Medical test must be made.

Bibliography

[1] Beatriz Perez Pardo, Bachelor Thesis: Localisation of game tokens in a digital display –Analysis and design of an autonomous game token system-, April 2009.

[2] http://www.ece.virginia.edu/~mv/edu/ee136/Lectures/week3.html

[3] Trehorel Jules, Game design : The RoboRally Board Game, August 2009.

[4] OPR2100, OPR2101datasheet from TT electronics Optek Technology

[5] http://www.xbitlabs.com/articles/monitors/display/lcd-parameters.html

[6] http://en.wikipedia.org/wiki/LCD

[7] Erwin M. Biebl, Sampling Architectures, 2003 IEEE

[8] Floyd M. Gardner, A BPSK/QPSK Timing-Error Detector for Sampled Receivers 1986

[9] Floyd M. Gardner, Interpolation in Digital Modems-Part I: Fundamentals 1993

[10] Lars Erup, Floyd M. Gardner, and Robert A. Harris, Interpolation in Digital Modems-Part II: Implementation and Performance 1993

[11] Fredric J. Harris, Michael Rice, Multirate digital filters for symbol timing synchronization in software defined radios 1998

[12] Fredric J. Harris, Michael Rice, Polyphase filterbanks for symbol timing synchronization in sampled data receivers 1999

[13] Fredric J. Harris, Michael Rice, Digital Receivers and transmitters using polyphase filterbanks for wireless communications. IEEE 2009

[14] C.A. French and W.A. Gardner, Spread-Spectrum Despreading Without the code, IEEE 1986

[15] Zhijin Zhao, Zheng Sun, Fei Mei, A threshold Detection Method of DSSS Signal Based on STFT

[16] José Manuel Andújar Marquez, J. Lluis Pijoan I Vidal, Diseäo y construcción de un sistema digital avanzado de comunicaciones en HF, Física de la tierra 2000

[17] A. Lee Swindlehurst, Normalized Adaptive Decision Directed Equalization, IEEE Signal processing letters vol. 5 NO.1 January 1998

[18] Jorge F. Schmidt, Juan E. Cousseau, Pedro D.Donäate, Ecualización con realimentación de decisión en redes inalámbricas.

[19]http://www.mathworks.es/products/filterdesign/demos.html?file=/products/demos/ship ping/filterdesign/adaptequaldemo.html

[20] David R. Pauluzzi, Norman C. Beaulieu. A comparison of SNR Estimation Techniques for the AWGN channel, IEEE transactions on communications, Vol. 48.NO 10, October 2000

[21] Yunfei chen; Norman C. Beaulieu, Maximum Likelihood SNR Estimators for digital receivers, IEEE 2005.

[22] http://java.sun.com/docs/books/tutorial/extra/fullscreen/

[23] http://en.wikipedia.org/wiki/Four_color_theorem

[24] Piotr Gajewsky, Channel assignment in CDMA systems with Walsh and PN coding, IEEE 2001.

[25] Lipo Wang, S Arunkumaar and WEn GU, Genetic algorithms for optimal channel assignment in mobile communications, Proceedings of the 9th international Conference on Neural Information processing (ICONIP'02), VOl. 3

[26] Andreas Gamst, Homogenous Distribution of Frequencies in a Regular hexagonal cell system

[27] http://spie.org/Images/Graphics/Newsroom/Imported/0016/16_fig1.jpg

[28] http://www.epilepsyfoundation.org/about/types/syndromes/reflex.cfm

[29] R. Neil Braithwaite, Using Walsh code Selection to Reduce the power Variance of Band-Limited Forward-Link CDMA Waveforms, IEEE 2000

[30] http://zone.ni.com/devzone/cda/ph/p/id/319

[31] http://java.sun.com/docs/books/tutorial/index.html

[32]Tran Van Muoi, Receiver design for digital fiber optic transmission Systems using Manchester Coding. IEEE 2005

[33] Zhan Jian, Wu Nan, Kuang Jingming, Wang Hua, High Speed all digital symbol timing recovery based on FPGA. IEEE 2005

[34] Matthew P. Donadio CIC Filter Introduction. Iowergian 18 july 2000

[35]Nidal S. KAmel and Varun Jeoti A linear prediction based Estimation of Signal-to-noise Ratio in AWGN channel. Etri Journal. Octover 2007

[36] Matthew trinkle, SNR considerations for RF sampling receiovers for phased Array Radars. University of Adelaide

[37] Hua xu, Hui Zheng. The Simple SNR Estimation Algorithms for MPSK Signals. IEE 2004

[38] Di Wu, Pedrag Spasojevic, Ivan Seskar, Orthogonal Variable Spreading Factor Codes with Zero-Correlation Zone fo TS-UWB,. IEEE 2005

[39] P.S, Moharir, Ternary Barker codes, IEEE electronics letters October 1974

[40] Axel Kohnert, Alfred Wasserman, Construction of binary and ternary self-orthogonal linear codes. University of Bayreuth 2008

[41] Jimmy Persson, Master's thesis: Methods of SNR estimation from Asynchronously Sampled Data in an 802.11b System. Lulea university of technology, 2004

[42] Norman C. Beaulieu, Andrew S. Toms, David R. Pauluzzi Comparison of four SNR Estimators for QPSK Modulations. IEEE Communications letters, February 2000

[43] Marvin K. Simon, Sam Dolinar, Improving SNR Estimation for autonomous Receivers. IEEE transactions on communications June 2005

[44] Markus A. Dangl, Jürgen Lindner, How to use a priori information of Data Symbol for SNR Estimation. IEEE signal processing letters, November 2006

[45] http://www.davidsalomon.name/DC2advertis/AppendH.pdf

[46] http://en.wikipedia.org/wiki/Color

[47] http://en.wikipedia.org/wiki/Lab_color_space

[48] Shaojun Xu, Daobem LI, Ternary Complementary Orthogonal Sequences with Zero Correlation Window, IEEE 2003

[49] Steven J. Nowlan, Geoffrey, A soft Decision-Directed LMS Algorithm for blin equalization, IEEE Transactions on Communications, February 2003.

[50]Michel Rice, Fred Harris, Loop control Architectures for Symbol timing Synchronization in sampled data receivers. IEEE 2002

[51] Kurt H. Mueller, Markus Müller, Timing Recovery in Digital Synchronous data Receivers, IEEE 1976.

[52] Chirs Dick, Fred Harris, Michel Rice, Synchronization in Software radios – Carrier and timing recovery using FPGAs. IEEE 2000

[53] Inmáculada Hernàndez Rioja, TEMA 2 Transmisión Digital Banda base

[54] Zujun Liu, Kechu Yi. Symbol timing synchronization using interpolation based matched filters. Springer published online: 7 November 2008

[55] Hong-Kui Yang, Martin Snelgrove, Symbol timing recovery using oversampling techniques. Carleton University

[56] http://courses.cit.cornell.edu/bionb441/Timers/index.html

[57] http://courses.cit.cornell.edu/bionb441/

[58] http://www.mathworks.com/access/helpdesk/help/helpdesk.html

[59] http://www.mathworks.com/matlabcentral/fileexchange/24654

[60] Robert C. Dixon, Spread Spectrum systems with commercial applications, Third Edition. Ed. John Wiey & sons

[61] Byeong Gi Lee, Seok Chang Kim, Scrambling Techniques for digital Transmission. Ed Springer Verlag.

Note for the appendix

In these appendixes but only in the copies given inside the Fraunhofer IIS will be written the program code. For the other copies only a brief description of the codes will be written. Fraunhofer IIS consider that this thesis can be public, but I also think that the complete codes will be only for Fraunhofer IIS

Appendix A

Java classes that are used with the Com.java and CombFrame.java, for setting on the screen the signal. This two classes have to divide the screen in a hexagonal cell array and then put a new image on the screen every time that the screen is refreshed. The new image is always the same background, some picture, and then every hexagonal cell change its brightness every refresh rate.

A.1 Movile_15_Stations_3_Cells.java

Creates the cells that I need for filling the hexagonal grid, and assign one to each hexagon.

A.2 Cell.java

Creates the signal which is an attribute of the cells instanced in the Movile_15_Stations_3_Cells.java. it has to set an orthogonal and different code to all the neighbours. And then it module the orthogonal code and build the frame. For building this, sometimes it has also to joint each one in stations that is why it has the static vector allstation, and assign a different and unique number to each cell when the class is instanced.

A.3 Auxiliaryprograms.java

This class make some operations that I need for an easy and not monolithic construction of the signal. Here are implemented the methods that modulate the orthogonal code, add continuous levels to the signal, and operates that I need with vectors of symbols.

Appendix B

In this Appendix I will write the Matlab programs that are used for demodulating the signal, this are only the most representative.

B.1 Capturesignal.m

It captures the signal, then filter and finally recovery the symbol timing for sampling it in the correct instant.

```
%% for loading the image
disp('load the image')
pause(10)
disp('begin')

%% channel parameters
fsa=48e3;
Nbits=16;
fsy=60;
N=1000;
tcapture=N/60;

%% build the channel and capture the first data
channel = audiorecorder(fsa,Nbits,1,1);
recordblocking(channel,tcapture);
```

B.2 Decodetostationused.m

It receives the sampled symbols, correlates with the different orthogonal codes, and selects the one that has more power after the correlation.

B.3 Fastreceiver_X.m

Is the final implementation of the demodulator in Matlab