LOCALIZATION IN A 3D INDOOR ENVIROMENT.

Author: **Federico A. Buratti**

August 2004

DUBLIN CITY UNIVERSITY
SCHOOL OF ELECTRONIC ENGINEERING

MASTER OF ENGENEERING
IN
TELECOMMUNICATIONS

Supervised by Prof. Paul F. Whelan

Acknowledgements

I would like to thank my supervisor Prof. Paul Whelan for the enthusiasm and commitment to this project. Thanks are also due to my tutor John Mallon and for his support and guidance to this work to date. Thanks go also to Dr. Ovidiu Ghita for his availability in giving me his support.

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed: ..... Date: .....

Abstract

The principle objective of this project is the measurement of an indoor environment by using the SICK Laser scanner. That means measuring the distance information of all the objects placed in front of the sensor in a way that is possible to recreate the same environment with the same distance characteristics as those obtained from the laser point of view.

The use of the Laser scanner requires a suitable design of a software / sensor interface that would allow the control of the device and an easier processing of the information received.

Using its working principle as a base is possible then to achieve more complex objectives; like joining a stack of singular scans to reconstruct a whole piece of environment linking then that information with a picture, is possible to join the volume information with the texture. The fusion of both gives to every point of the graph two kinds of information; one related with the distance from the point of view, and one related with the texture that will give colour to it.

The final aim of the project is an experiment that shows how is possible to use the information given by the laser and the information given by a digital camera in order to obtain the spatial information between both sensors.

Table of contents

Table of Figures

CHAPTER 1

1.1 Introduction

One of the main parts of the project is the development of a software interface that puts in communication the laser and the user. The interface is organized in a layered structure where all the upper layers make use of the functions of the lower ones and process the results obtained, the base layer is the one that has the task to read and write to the port, the device layer on the other hand has to form the packets to be sent and has to check their correctness. The development of the interface means also the design of a suitable graphical user interface that allows the easy control of the device and a clear view of the data received by plotting it and creating a graph representing the scan performed of the environment. In that section several useful mechanisms are included, all of them are part of the top layer so they all work on the data to be graphed modifying it for a more suitable view (magnification, displacement on the graph etc.).

Among the objectives of the project is the 3D representation of the environment, that means a recreation of a scanned scene by using the data received from the scans for mapping the depth, in particular three matrices are created representing the x, y, z information, meshing them together is possible then to recreate the shape of the object.

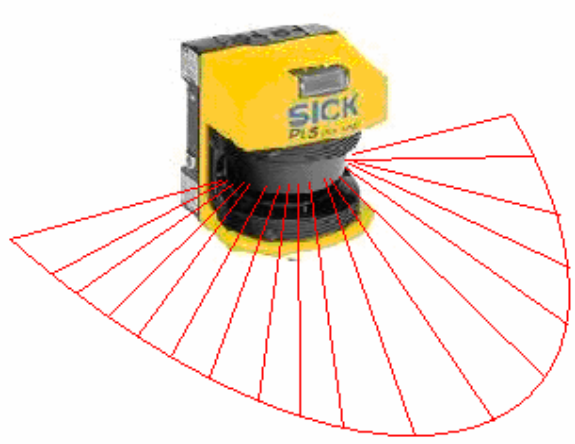
At that stage is possible to work with the created matrices, in particular is possible to use the z matrix to see all the distance information highlighted with grey scale colours as an image.

Is now possible to add the picture information, that means the texture information given by a camera. The picture will be taken from a different angle than the laser point of view. A mechanism is needed in order to be able to find the location of the camera relative to the scanned data. Using this information the 3D data can be transformed into the viewing angle of the camera. A token point matchery scheme is used to calculate this transformation. The important thing to take in consideration is that a considerable number of points are needed in order to get reliable results, in our experiments we considered the corners of the objects, so by using a scene with a considerable number of corners is easier to locate and match them.

At this stage is possible to add texture to the 3D representation, by placing over it the image, that has been scaled before, that should now match with the most relevant points.

CHAPTER 2

2.1 The laser scanner (Working principles)



2.1.1 Physical principles:

The laser is also called Time of flight laser because its measurements are based on the time the laser beam takes to reach the object surface and to come back to the sensor. The time elapsed will give information about the distance of the object.
By using some rotating mirrors the laser beam will be sent to all the area around for 180 degrees [1].

2.1.2 Operating principles:

The Laser scanner communicates with telegrams, that means that all the information received by the user and the information given back must be inserted into data packets.
The length of the packets can change depending on the kind of packet it is.

Among the kinds of packets that can be sent by the laser there is the start up message, that is sent after few seconds the laser is turned on, at that moment the device is able to go to any operating mode.

The two operating modes used in the project are the "*installation mode*" and the "*averaged scans mode*". In order to choose one or the other the correspondent packet must be sent by the user.
The "installation mode " has the characteristic of performing a complete scan of 180 degrees with a resolution of 0.5 degree, the maximum distance achieved is 80 m, the advantage of this mode is the good resolution, suitable for a detailed measurement; but a disadvantage is the unuseful information if we just want to focus onto a specific object of the environment, for this reasons some filters must be designed in order to consider just the part of interest.
The "averaged scans mode" is quite different than the previous one. Firstly it gives the possibility of choosing the scan sector, that means that is possible to choose the start and stop sample number in

order to just consider the desired area of the scan. The disadvantage of this operating mode is the resolution, in fact in this case each sample is taken every 1 degree and an averaged value for each one is returned. [1]

After sending one packet for choosing the most suitable operating mode, an acknowledgment packet is received back from the laser, is now possible to start scanning in one of the two ways detailed above. The first packet will be an acknowledgment and the following will be scans performed continuously.

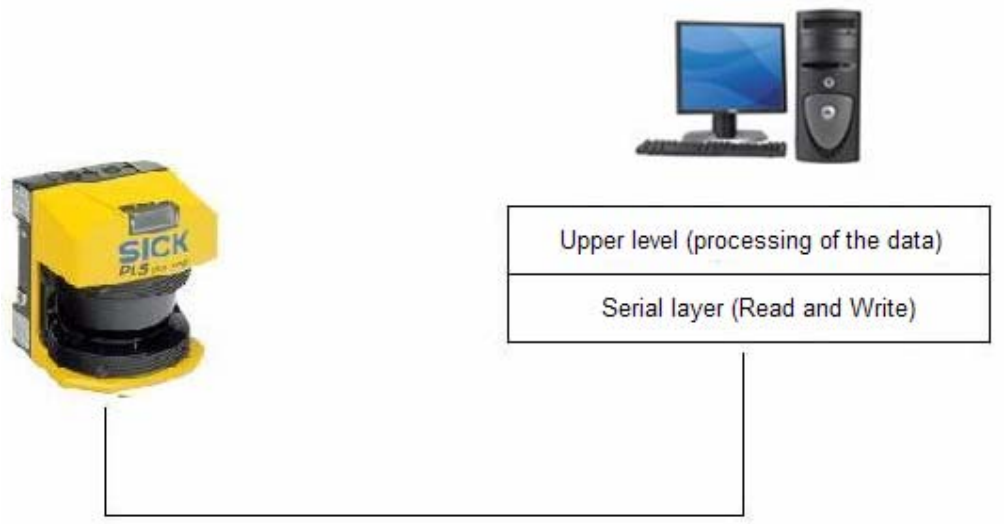
CHAPTER 3

3.1 The interface

As explained in the previous chapter, the laser communicates by packets, it changes its operating modes when receives specific packets and sends data embedded in packets, so in order to make the communication user / laser easier a suitable interface must be designed. The main objective of this is to send the respective data representing the user operation choice and to receive and automatically display the useful part of every packet.

The interface is organized in a modular way, the structure is divided in different layers starting from the base level (used to read and write to the port) to the top level, which is the laser itself with its several function that make use of the data provided by the lower levels.

In picture 1 is displayed the flow chart of the interface.



Pict 1 . Structural diagram of the interface organization

Picture 1 depicts a general view of how the communication between the user and the laser is performed. By using a layered structure each layer uses the service given by the lower one, in this case the serial layer is the base and has the task of communicating with the laser, that means sending the data to the port and reading from it. The data sent to the port is formed by requests made by the upper level that must be delivered to the laser in some way, while the data received is in general formed by packets containing the values representing the scans of the laser.

3.1.1 Serial Layer

The Serial layer is initialized every time the interface starts, all the parameters used for the communication are set up:

```
Serial Constructor
[
      .
      .
hCom = CreateFile (PcCommPort)        // Set up of the Com port communication;
Timeouts (Read , Write) setup;
Parameters of the Communication (Byte size, Stop bit, Baud Rate);
Start the Reading Thread();           // Starts reading the port
      .
      .
]
```

The reading function:

The read function is implemented by using a thread loop that is continuously waiting and checking for new data arriving to the port.

When the thread starts, it receives the pointers to the serial communication created before.

```
Reading Thread
[
      .
      .
      .
Receiving pointer to Serial communication; // will read the same port that the
                                              the Serial layer writes to

Declaration and set up the OVERLAPPED structure (StructureName.hEvent);// Event
                                    parameter that is set every time new data
                                    is coming. Needed later for reading and
                                    writing  in the same port and at the same
                                    time.

Loop Continuously
[

if (Wait (StructureName.hEvent)) //The loop will remain at this point until the
                                   Event parameter will not be set.
                                   If the event occurs new data has arrived.
[
Read Values;
]


While (value received != 02)    // The loop will keep receiving data until the
[                                   start bit has not been encountered.
Read the next one;                  Used to keep the length of the header of the
                                    packets always the same.


If (first value received)
[

Read (Temp);      // Read all the values and place them in a buff called temp
Length = Temp (ind) // The length of the packet is found in the "ind" position

]

]
```

The Overlapped structure displayed above in the code allows the use of the port for reading and writing at the same time. If such a structure wouldn't have been used the loop would have been remained staked on the port waiting for new bits to arrive, that means that the door would have been kept permanently busy making it impossible to write on it by the Serial layer. For that reason is employed such a structure that allows the port to be used for writing and reading at the same time, it is called Overlapped because allows the asynchronous input and output in the interface.

The overlapped solution makes use of an event handle that every time a new I/O event occurs it change the state of a member of the structure, then by just setting up an event handler to that member is possible to determine whenever new data has arrived to the port.
With this solution is possible to leave the process waiting for a different object rather than directly for the event on the port, this would keep the port free for all the time allowing the writing on it whenever it might occur.

Whenever an event occurs new data has arrived, at this point the first bit received must be checked in order to keep the length of the header always the same, that means starting always from the same value. The header must always be the same length because in order to get the length of the entire packet for the reading always the same bit position must be read (in the pseudo code above that is the position indicated with "ind.").

The length value found is very important because will be used even later for proving the correctness of the packet, another purpose of it is the "cleaning" of the packets, that means getting rid of all the bits that don't represent information and just fill the buffer with the ones that will be used in the upper layers.

When the data representing useful information has been found by the mechanism above, must be placed in the buffer to be delivered to the upper layer. Since the read function is continuously receiving data (during a scan) the temporal buffer must be locked in order to make sure that the buffer will not experience changes during the copy.

```
EnterCriticalSection; //lock the local buffer
Buffer -> temp        //performing the copy of the buffer
LeaveCriticalSection; //unlock the local buffer
```

3.1.2 Upper layer

The upper level is the layer that makes use of the service of the lower one. The main objective of this level is the creation of the packets to be sent to the device, this means the formation of the data packets that correspond to specific messages for the laser.

The installation packet will be formed as following:

```
Install message()
{
buffer = [1010101001…….]   // The buffer is created
Serial - > Write (buffer)   // The buffer is sent to the lower level to be
                               written to the port
}
```

Another important task of this layer is the calculation of the check sum, this mechanism is needed for proving the correctness of the packet received. Since the port is receiving scans continuously when the most recent scan is wanted the program will check its correctness before; if the value of the check sum returned by the function is the same as the one in the buffer the packet has not been corrupted and can be sent to the upper layer to be graphed.

```
Get last scan
[
if (Check sum calculated = CRC of Last scan)    // Checking correctness
NewDataBuffer = Last scan                       // Copying the buffer
]
```

Once the data packet has been tested for errors, a new buffer is then created containing the most recent scan, this buffer is then sent to be graphed in the top layer.

The final stage of the communication is the graphical user interface, that requires the development of a software capable of making easy the control of the interface by the user and able to graph the data received.

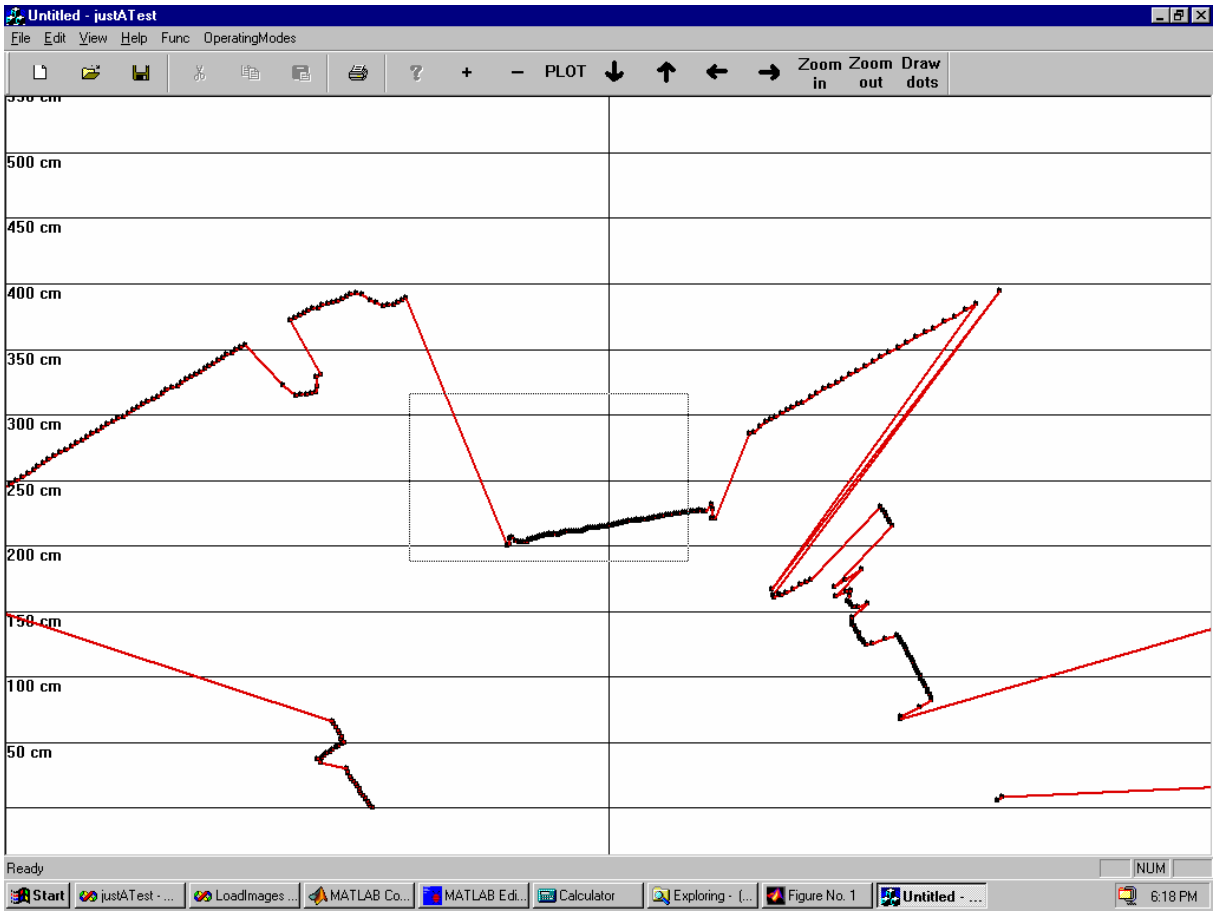
3.1.3 The software:

This is the nearest level to the user, it provides all the tools needed to control the interface. The data contained in the packet coming from the lower level will be used to make the graph. The values received are represented in the graph as the distance from every point to the center, so in order to get the x and y coordinates the angle must be used. Once all the coordinates of every point are found, is possible to join everyone with a line creating this way the graph of the scan.

A graphical window was developed in order to help the user to have good control of the data, from there is possible to display the created graph of the last scan, is possible to move around it, is possible to zoom on the complete image, is possible to maximize the selected area and is possible to highlight the points on the lines representing the samples received and used to draw the uniform line.

Following is a picture with the graphical user interface of the software created, in the picture is showed a graph of a 180 degree scan, it will be formed by 360 samples since its resolution is 0.5 degree. The graph is drawn over a grid that represent the real distance of each point from the sensor.

The buttons showed represent the zoom that simply redraw the graph by dividing or multiplying every distance value by a scale factor, the arrows are needed to move around in the graph, they just displace the origin by a certain amount in the desired direction, the "draw dots" button is used to insert and delete the dots in the graph where each one of them represent a scan sample.



Pict 2 . A shot of the developed software

Picture 2 represents a typical graph of a scan taken from the laser, the laser is placed in the point where the vertical and horizontal line cross each other, in the image is showed the rectangle used to select the area to enlarge, in that case it graphs a corner of the wall. In the image are displayed the dots representing where the samples used to create the graph are placed over the line, the algorithm used for the creation of that graph is detailed in the next chapter.

One other feature is the possibility to select the sector of the scan by just giving the start and stop values in a dialog box. In this last case the values obtained would be less since a smaller sector is chosen and because of the resolution would be of 1 degree.

CHAPTER 4

4.1 The scan

Once the laser is set to the scan operating mode, it makes scans continuously and sends them to the user, since each 180 degrees scan is contained in one packet, a sequence of packets with the same length will be received.

The scan information contained in every packet is formed by 360 values that represent the t.o.f of each sample (i.e. the distance between the object encountered and the sensor).
In order to have a better view of one scan, is possible to make a graph of it representing the upper view of the shape of an object. In particular with these scans, is possible to have an idea of the profile of the object as seen from the top.

The first step needed to achieve that purpose is the calculation of the x and z coordinates for each sample. For that purpose the angle between the laser beam and the vertical is needed.

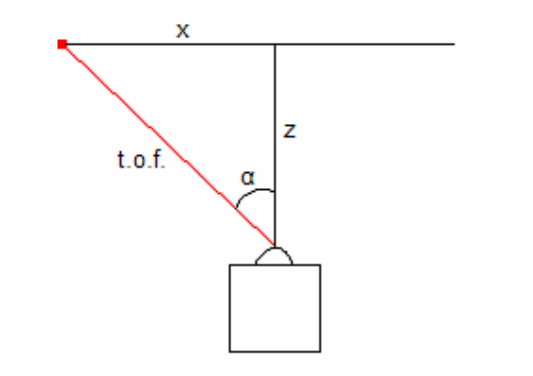


Figure 1 . Two dimensional decomposition of one sample of the scan

In figure 1 is showed how the x and z coordinates are found:

$$z = Cos\ \alpha\ .\ t.o.f$$
$$x = Sin\ \alpha\ .\ t.o.f$$

Since the laser scan starts from one angle we consider to be 0 and goes all to 180, is possible to calculate the x and z coordinates and find the point on the graph every time a new value has been read from the sequence.

```
angle = 0;              // At the beginning the angle is considered to be 0;
starting point = (x,z)  // The first value of the sequence is read and graphed
loop (360)     // The process will loop until all the points have been graphed
[
Get next value()        // The next value is read
z = cos (angle) . next value;  // Calculating the coordinates of the new point
x = Sin (angle) . next value;
Draw line to point (x,z);   // Connecting with a line the previous with the new
angle = angle + 0.5;        // the angle is increased of .5 since that is the
                               scan resolution
]
```

Figure 2 is a drawing that shows how a graph of a scan is performed; by starting from the first point and moving on finding, on the z, x plane, all the successive points is possible to create a shape like the one showed (it can be a room or a singular object).

In particular is possible to see from the graph that the z coordinate of each point gives the distance from the sensor in the vertical line, while the x gives the distance in the horizontal one.

The image shows two samples taken at different angles used to build the graph, as is clear in the picture when the angle formed between the laser beam and the horizontal is bigger than 90 degrees, the complementary one is used and the sign of the x value is changed.

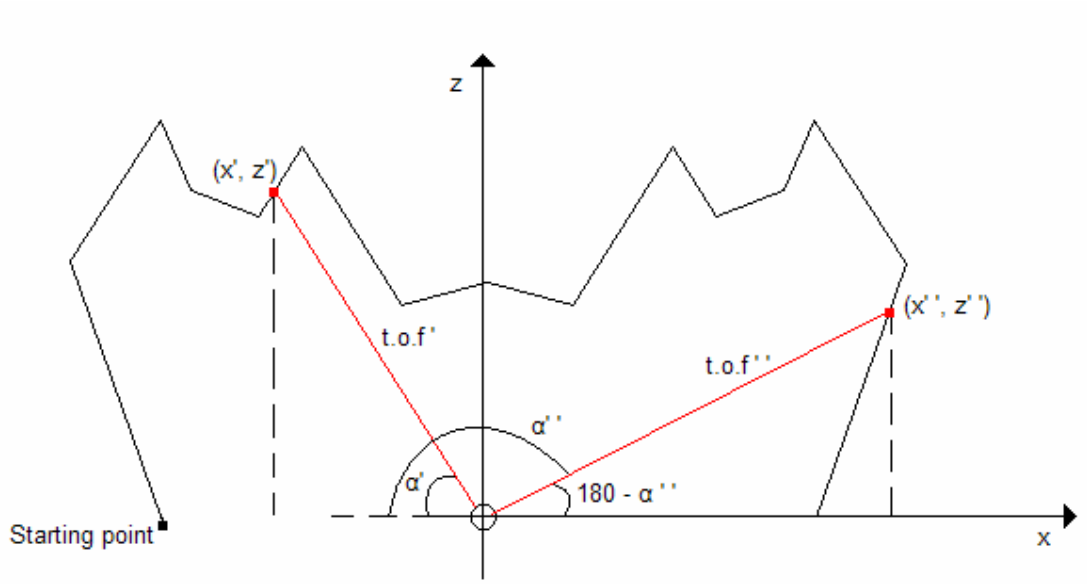


Figure 2 . Graph of a complete scan

At this stage we are able to join more singular scans in order to have a view of the space, the idea we have to follow for this purpose is to take one scan from a certain height, then take a new scan from a little bit higher point and so on until the whole object has been scanned. Is true that with this solution the more little is the amount raised the more accurate will be the result.

Once a considerable number of scans have been obtained, is possible to recreate the scene by displaying them in a stack with the same order as they were taken, this way is possible to get the whole profile characteristics of a certain scene examined.

4.2 3D Reconstruction

As exposed in the previous paragraph the scan of an object must be performed with successive scans at increasing height; by following the same principle is possible to get a complete scan even just rotating the sensor but keeping the laser always at the same height.

Since for recreating an object a stack of scan must be taken from the bottom to the top, an handier way to do it is by rotating the laser, always facing the object, increasing its angular position of a bit every scan performed; even in this case the smaller is the angle increased every time, the more accurate will be the results.

In order to make this solution working we have to consider the angle formed between the laser beam and the vertical, in figure 3 is displayed what angles are considered and what is the mechanism of the measurement.

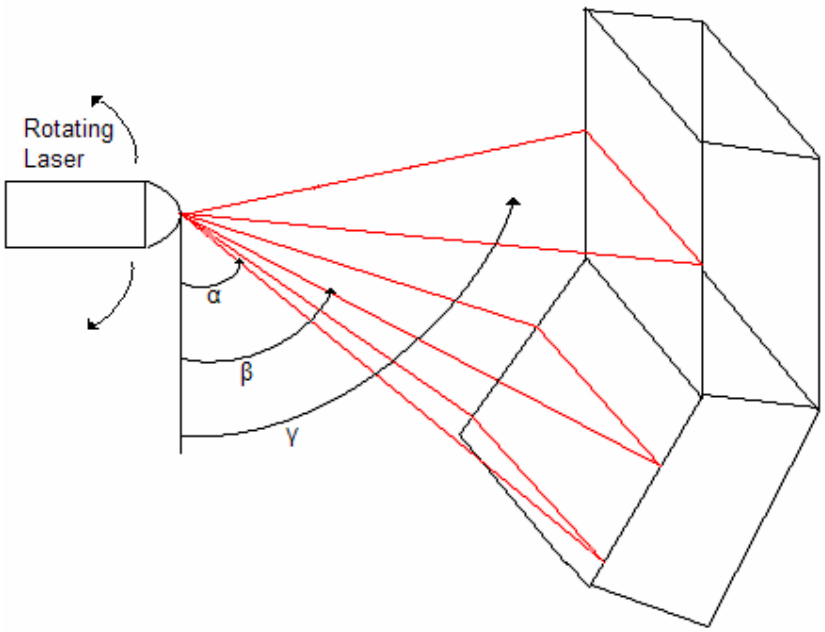


Figure 3 . Scans performed to get the spatial information of the object

Since in this way the laser will be always at the same position, the distances revealed will not be the real ones due to the inclination. This problem can be solved by considering the angle between the beam and the vertical, by the mathematical function is possible to calculate the value of

the distance from the object to the laser like if the scan were taken from the same height (z coordinate in figure 4).

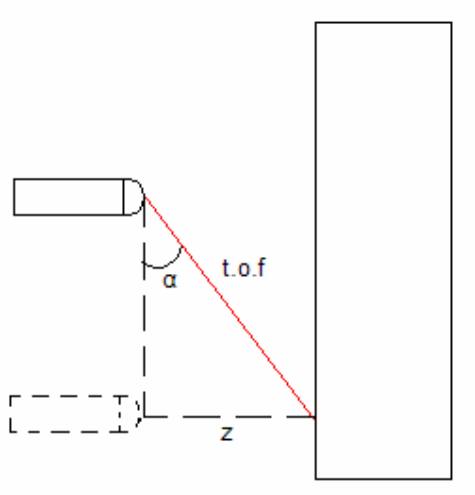


Figure 4

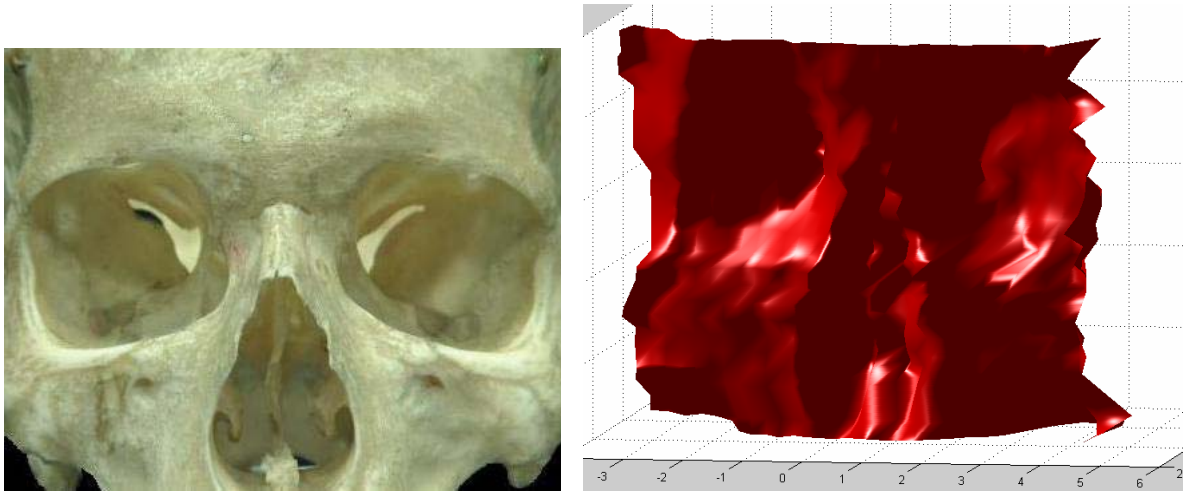
$$z = Sin(\alpha) \cdot t.o.f$$

The value found for z is the same as if the scan were taken by the dotted laser displayed in figure 4. Next calculations needed to get the x coordinate, is the same as the one explained previously in the case of the 2D representation.

Since we are now trying to build up a 3D model of a scene, we need the three coordinates in the space; until now we have found x (that is the width) and z (that is the depth). Is possible to get the information about the height by just supposing that every scan is taken from every unit of y.

In our experiment we used this kind of approach and by increasing the angle of 1.5 degrees every scan is possible to get reliable results.

The last step for performing a 3D scan is the data organization. For that purpose we created three matrices, one for each coordinate information. This matrices are organized in a way that every row represent a scan and every column is a sample of the scan. For every value in the x corresponds a value of y and z placed in the same position but in their respective matrix.
Once the three matrix are set up, is possible to graph the model by meshing the values. Picture 3 displays some pictures relating to a test performed with a skull using this kind of mechanism.



Picture 3 . The skull as it is seen from the laser point of view a) and its 3D model created with the laser b)

By using the Laser scanner with the skull depicted above, we are trying to test the reliability of the data referring to the distance values. That means we are trying to test what kind of representation we could get of an object with many different depth levels.

Depicted in the picture 3(b) is the 3D model of the skull obtained, as clear in the picture the three mayor holes of the eyes and the nose were revealed and plotted in the graph.

The resolution obtained is not brilliant even because we are working with a relatively small object in respect to the resolution of the sensor, but the result obtained can still be considered satisfying.

CHAPTER 5

5.1 Mathematical formulations

Once the scan has been performed and a complete set of x, y, z values have been created, is possible to add the information given by the picture. That means trying to find the matches between both sets of information, in other words trying to find the same data in both representation and from that calculating the displacement performed.

The principle that is at the base of the experiment is related with the way of how the location of the points of an object in a 3D space changes after a displacement is performed; that means, by knowing the original location (x, y, z) of every point, is possible to know what the final location will be by knowing the amount of rotation and translation performed by the object (both these information are considered as an individual matrix).

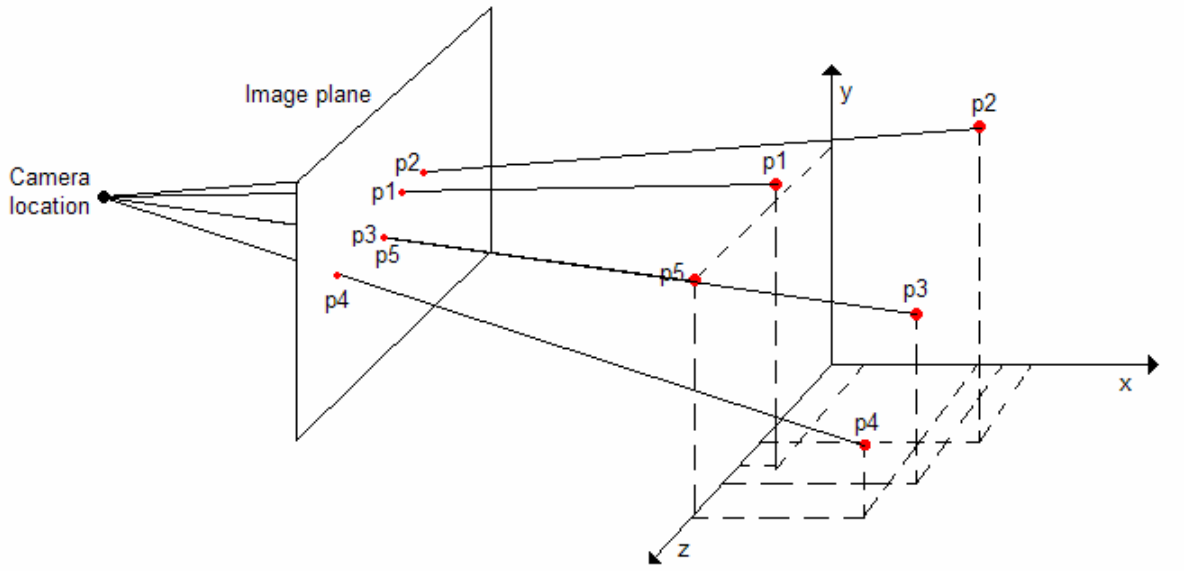


Figure 5 . Set of points in the space with their correspondents in an image

In figure 5 is depicted how some point in the space are represented in a bi-dimensional plane as the one of an image.

At this stage, our goal is to find out the new position occupied by every point in an image plane after a rotation. In fact by knowing their space coordinates and making use of the matrices representing the angle, the direction of the displacement and the camera intrinsic parameter, is possible to set up an equation that gets the result.

The problem can be solved by using the following mathematical formulation [2]:

$$P = K \cdot [R \mid T] \cdot W$$

where the matrices have the following shape:

$$P = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \quad ; \quad K = \begin{pmatrix} fx & 0 & u \\ 0 & fy & v \\ 0 & 0 & 1 \end{pmatrix} \quad ; \quad R|T = \begin{pmatrix} r1 & r2 & r3 & tx \\ r4 & r5 & r6 & ty \\ r7 & r8 & r9 & tz \end{pmatrix} \quad ; \quad W = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

P is a vector of three elements representing the new coordinates of the point after the displacement. R|T is a 3x4 matrix representing the rotation performed, it is calculated by using the three matrices representing the rotation for each one of the axis multiplied then together [3], this means:

$$R = Rot(x, \phi_x) \cdot Rot(y, \phi_y) \cdot Rot(z, \phi_z)$$

The last column of the matrix represents a vector indicating its three coordinates that gives the amount of translation executed (T). W is a vector of four elements representing the original coordinates of the points in the space.

Another important principle we have to consider for our experiment is related with the insertion of the camera, in other words tells the way how the points in the space are located in the picture. For that purpose we have to make use of a matrix representing the camera intrinsic parameters called K

The values in the K matrix represent the focal length of the camera (*fx, fy*) and the coordinates of the principle point (*u, v*), that in general is the center of the image [2], but in my experiment is considered for simplicity to be (0,0) , for that reason all the pixel coordinates will be shifted back to the top-left corner.

$$Xpix = x - \bar{x}$$
$$Ypix = y - \bar{y}$$

Where $\bar{x}$ and $\bar{y}$ are half the image size.

The resulting 3x1 vector P contains the coordinates of the point in the image, stating that for the image just two coordinates are needed, in order to get the x and y values we divide every one by z getting a vector like the following.

$$P = \begin{pmatrix} x'/z' \\ y'/z' \\ 1 \end{pmatrix}$$

In this way is possible to find on the image plane the correspondent point with coordinates given by (x'/z', y'/z').

5.2 Development of the experiment

At this stage the formulas at the basis of the problem have been introduced, is now possible to find a certain algorithm in order to achieve the idea of the project. In particular is possible to think about my test as the scan of a particular object taken from a specific point of view and a picture of the same object taken from a slightly different angle.

The situation we are going to work on is the one depicted in figure 6.

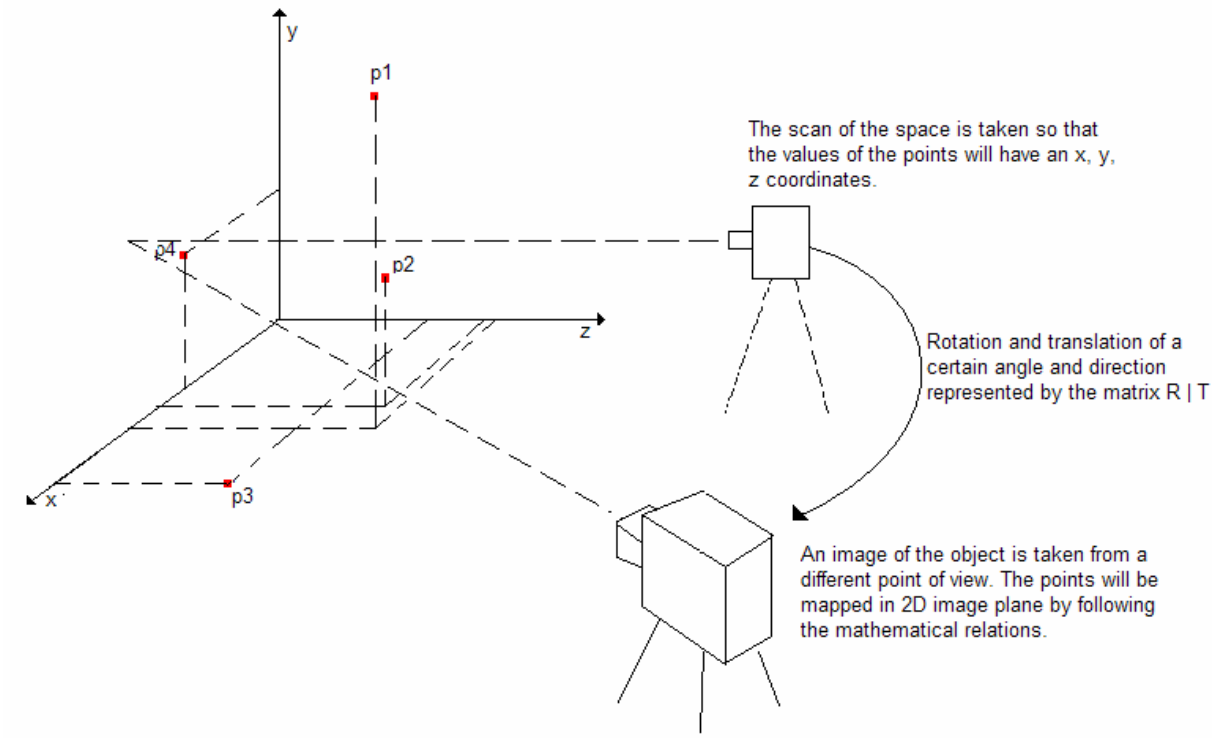


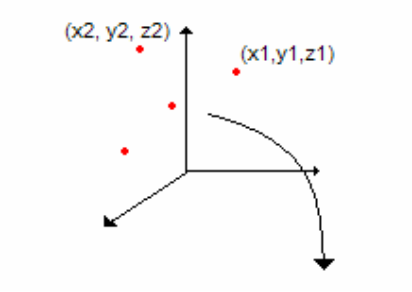
Figure 6 . Situation of my experiment

As is depicted in figure 6 we are using two kinds of sensors that will process the data in different ways, in fact the laser placed just in front of the object will get the whole space information of the points, that means is able to locate a point perfectly in the space by knowing its three coordinates, the camera on the contrary is not capable to map the information related to the distance from the sensor, for that reason some sort of approximations must be taken in the developed mechanism. Following is a detailed explanation of the algorithm and all the most important pseudo code involved.

Step 1

The first stage of the process is the scan by using the laser, it will return the whole space information of the points. The data obtained are then organized in matrices, more exactly three matrices will be generated, one for every axis.

The scheme in figure 7 represent the organization of the data of N scans of 360 samples each.



$$ScanX = \begin{pmatrix} x1,1 & x1,2 & . & . & x1,360 \\ . & & & & \\ . & & & & \\ . & & & & \\ xN,1 & xN,2 & . & . & xN,360 \end{pmatrix} \quad ; \quad ScanY \quad ; \quad ScanZ$$

Figure 7 . The organization of the data from the scanner

The three matrices will have the same dimensions since for every value in one matrix correspond another one in each one of the other two, and exactly at the same position. The values at the position *(ScanX(i,j), ScanY(i,j), ScanZ(i,j))* locate one point in the space.

Step 2:

Next step makes use of the picture from the camera, is now when the same group of points must be sought (anchor points) in both laser and image representation. Since a system of a considerable amount of equations must be solved, the same quantity of anchor points are needed.

For that purpose is used the z matrix displayed in a grey scale image, in such a representation the difference between distance values are graphed in different gray scale colors, the final picture displayed will have the same shape of the object and will be used to find and match the anchor points.

```
Display ScanZ
[
map = gray(256);     // Setting up the colormap that will be used for drawing in
                        the image
image(ScanZ, map)    // Displaying ScanZ using the defined colormap
]
```

5.3 Point matching

Once the z matrix has been displayed is possible to proceed with the point matching, that means finding out the coordinates of the same set of points in both images.

This process is performed by finding out and matching manually all those points that are referred to the same part of the object in both representation. In my experiment we considered the corners as the anchor points to be sought, so by opening the two images at the same time we created a list of all the coordinates of a considerable number of corners. The process is illustrated in figure 8.

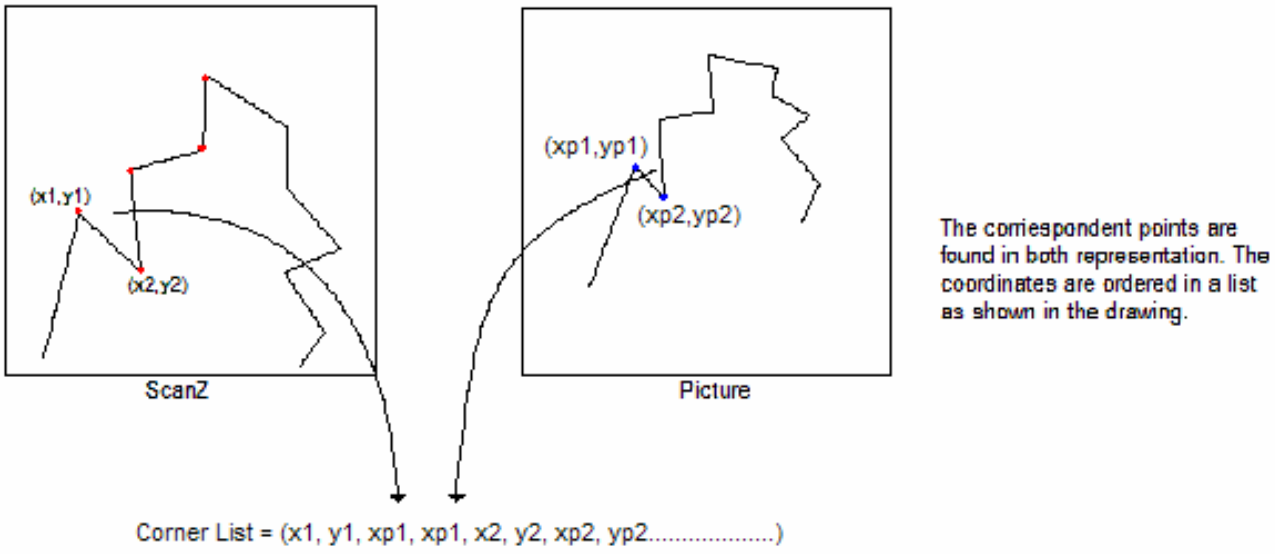


Figure 8 . Creating a list with the coordinates of the anchor points

Considering the to location of each point in the z matrix, as drawn in figure 8, is possible to get the X and Y values of the same point, by just looking up in the x and y matrices at the same location.

```
For (i=1 -> Numb of points)
[
matrix (1,i) = ScanX(x1,y1)     // Looking up for getting the x value
matrix (2,i) = ScanY(x1,y1)     // Looking up for the y value
matrix (3,i) = ScanZ(x1,y1)     // Looking up for the z value
]
```

Once all the list of coordinates have been completed, is possible to proceed by applying the formulas to this data.

Remembering that :

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Where the 3x4 matrix is the product of KR|T that multiplied by the space coordinates of the original point is possible to get the coordinates of the one in the displaced representation.
By developing the multiplications I will get the following equations:

$$x' = ax + by + cz + t_x$$
$$y' = dx + ey + fz + t_y$$
$$z' = gx + hy + iz + t_z$$

Since we are trying to locate the points in a 2D plane, the z coordinate must be set at one, so by just dividing x and y by z we will obtain

$$x_p = x'/z' = (ax + by + cz + t_x)/(gx + hy + iz + t_z)$$

$$y_p = y'/z' = (dx + ey + fz + t_y)/(gx + hy + iz + t_z)$$

$$z_p = 1$$

The relation above can be rewritten in the following way by using matrices

$$\begin{pmatrix} x & y & z & 1 & 0 & 0 & 0 & 0 & -x_p x & -x_p y & -x_p & 1 \\ 0 & 0 & 0 & 0 & x & y & z & 1 & -y_p x & -y_p y & -y_p & 1 \end{pmatrix} \cdot \begin{pmatrix} a \\ b \\ c \\ t_x \\ d \\ e \\ f \\ t_y \\ g \\ h \\ i \\ t_z \end{pmatrix} = 0$$

The first matrix is a 2x12 and is multiplied by a vector of length 12, every anchor point found will add 2 more lines to the matrix, so that to create a bigger matrix with 12 columns and a number of rows equal to the double of the number of anchor points.

```
For (i = 1 -> N. of points)
[
Get (x, y)        // Getting the x, y values from their matrices using the
                        coordinates obtained from the list
A ((i*2-1),:) = …  // Filling the first row as showed in the matrix above
A ((i*2),:) = …    // Filling the second row
]
```

5.4 The solution

At this stage we have to solve a matrix equation that is equal to zero of the form:

$$Ax = 0$$

Where A correspond to our known factor since is composed by all the coordinates of the anchor points founded previously. On the other hand the matrix x is our unknown since is the vector containing all the values of the K * rotation | translation matrix that we want to obtain.

One way to solve the showed matrix equation is by calculating the eigenvalues of A, finding the location index of the minimum of them and getting the whole column correspondent to that index in the eigenvectors matrix. The vector obtained correspond to x and can be replaced in the equation above for a proof.

```
Obtaining x
[
n = eigenvectors (A) //n is a matrix where the column are the eigenvectors of A
m = eigenvalues (A)  // m is a vector containing all the eigenvalues of A
ind = location (min (m)) // ind will be the index of the smallest of the
                              eigenvalues
 x = n (column (ind)) // x will be the eigenvector located at ind position in
                              the n matrix
]
```

When *x* has been found we have obtained an array containing the values of the 3x4 matrix K* R | T, so composed by the translation values tx ty tz as well. In order to get the two factors separately we can create a matrix just formed by the K and R values.

That means placing the values in the matrix without considering the 4^{th} , 7^{th} and 12^{th} value of the eigenvector considered.

If m is the eigenvector the pseudo code will be as following:

```
K*R = [m(1,ind) m(2,ind) m(3,ind);m(5,ind) m(6,ind) m(7,ind);m(9,ind) m(10,ind)
                                    m(11,ind)];
```

Since in the matrix just created is included the camera intrinsic parameter K, we have to make sure to separate it from the rotation matrix.

Remembering that K is considered as a 3x3 matrix where all the values are zeros except the diagonal, is possible to decompose the K*R matrix in three factor matrices, where one of them is a diagonal matrix that will be considered as our K and the other two will form our rotation matrix

In particular $K \cdot R = U \cdot S \cdot V'$

Where:

$$R = U \cdot V'$$

$$K = S$$

Once the R matrix is found, is possible to obtain the rotation angle by using the following formula [4]:

$$R = \begin{pmatrix} r1 & r2 & r3 \\ r4 & r5 & r6 \\ r7 & r8 & r9 \end{pmatrix} \quad \tan\phi = \frac{\sqrt{(r8 - r9)^2 + (r3 - r7)^2 + (r4 - r2)^2}}{(r1 + r5 + r9 - 1)}$$

The meaning of the displacement angle of the camera that we just found, can be thought as the rotation angle of the object in the space. That means that the object has been rotated of a certain amount in a certain direction in order to reach a certain position. The object from this new position will be seen from the laser point of view the same way the original model were seen by the camera. This concept will be detailed more in the next chapter with an example.

The equation outlined above will return an angle φ that is the composition of the rotations by the angles respect to the three axis x, y and z. By mathematical formulation is possible to define as well the vector that gives the direction of the global rotation angle φ found.

5.5 Simulation of the algorithm

In order to prove the algorithm outlined above we have performed a sort of simulation by using some random generated data to build up our 3D original object. In our simulation we create previously our K and R matrix by using a random angle value; multiply that matrix obtained by the data originated, we can obtain the coordinates of the same points placed in the image taken from a displaced location.

```
K = [600 0 0; 0 700 0; 0 0 1];//Setting up all the matrices needed for the sim.

Rx = [1 0 0 ; 0 cos(10) -sin(10) ; 0 sin(10) cos(10)];
Ry = [cos(10) 0 sin(10); 0 1 0; -sin(10) 0 cos(10)];
Rz = [cos(10) -sin(10) 0; sin(10) cos(10) 0; 0 0 1];
R = Rx * Ry * Rz;

P = K * R ;
```

The rotation matrix in our simulation is considered of dimension 3x3 (i.e. the translation is not considered)

```
for t = 1 : 100          // Generating our random data and placing the
   p_w(1,t) = rand * 10        // values in a 3x100 matrix
   p_w(2,t) = rand * 10
   p_w(3,t) = rand * 10

end
```

```
p_1 = P *  p_w;                 // Multiplying the data by our P matrix

p_1(1,:)=p_1(1,:)./p_1(3,:); // Converting the data obtained from the space
p_1(2,:)=p_1(2,:)./p_1(3,:); // to the plane, dividing by the z value each time
```

Once the data in both space and image has been found and organized in matrices is possible to continue by setting up the A matrix, formed by all the coordinates of the points found.

Since in our simulation we are not considering the translation our KR matrix will be of dimension 3x3, so we have to set up a matrix A with 9 columns and 200 rows, since we are considering 100 points in the space.

The aim of our simulation is to check if the K'R' matrix that we will be able to find by using the matrix A created and following the mathematical procedure is similar to the matrix P set up at the beginning of our experiment. In other words we are checking if we would have got the same rotation matrix KR if we just had known the location of the points in the space and in the picture (as will be the case in our final test).

Continuing with the following step we can obtain the value of K' and R' matrices separately by using the eigenvector theory.

The last step is to prove that the K'R' matrix found by mathematical formulation is the same as the original one that has been used to obtain part of the data for the experiment. One way to do our test is by multiplying the original data by the K'R' matrix found, locating and displaying the points in a 2D plane. Creating the same plane but this time using the original KR matrices and graphing the picture obtained in the same graph as the previous is possible to verify the quality of the algorithm designed.

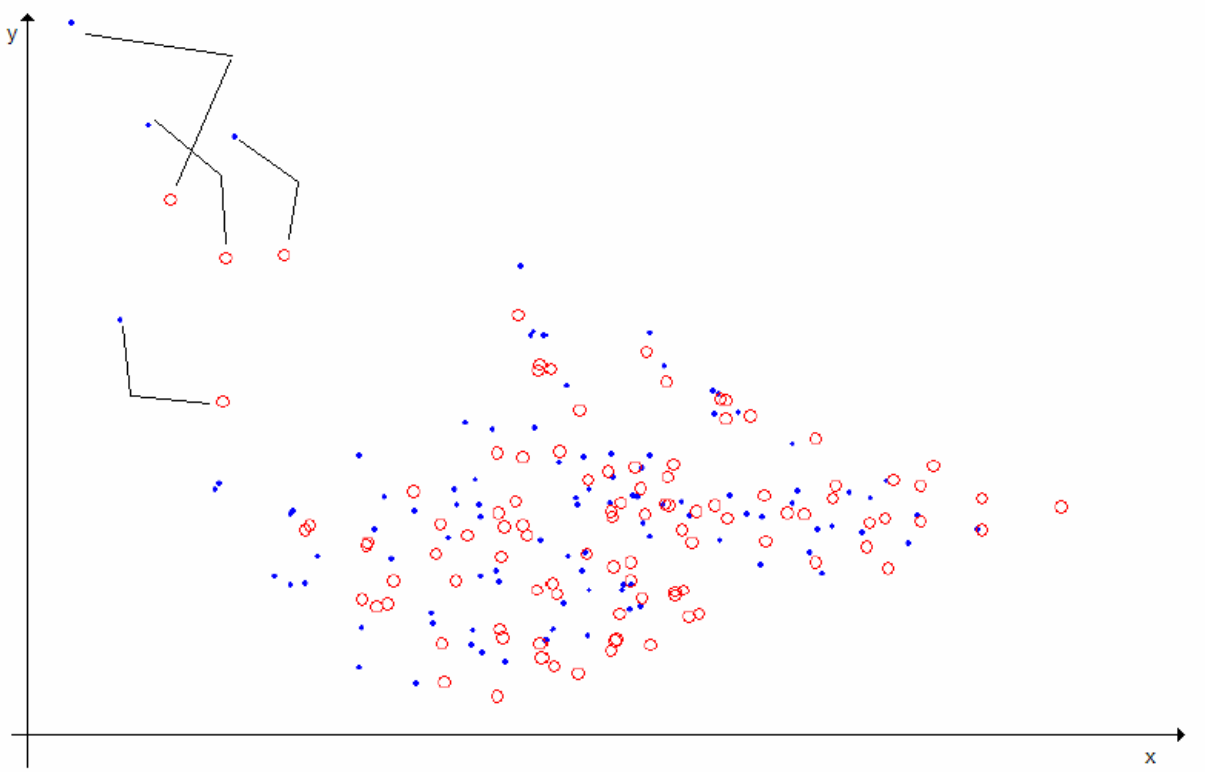


Figure 9 . Distribution in the image for the points found with KR and K'R' rotation matrices

Figure 9 displays the results obtained, as is clear the points obtained are roughly at the same position. By just fixing the attention on those signed in the figure is possible to observe that the respective location of each point with the others of the same group is the same, means that both clusters have the same internal distribution of points. From the graph is possible to realize also that both clusters are well overlaying one over the other, that means that the point coordinates found with the estimated rotation matrix and those found with the original matrix are roughly the same, this fact prove the correctness of the algorithm developed.

CHAPTER 6

6.1 Test and results

As a test for the algorithm explained above we created a scene formed by boxes, these were chose in order to get a considerable number of corners.

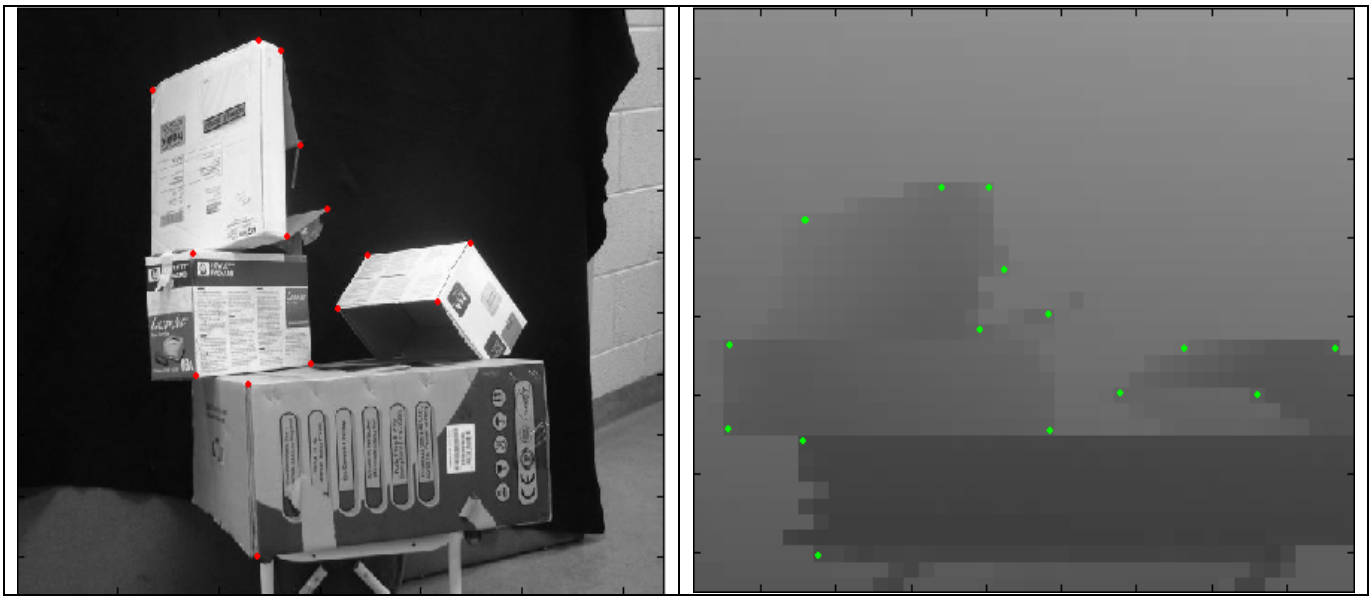
A scan of the boxes were taken from the front and a picture were snapped from a slightly moved angle.

By displaying the z matrix obtained with the scan is possible to proceed with the point matching. This step of the procedure requires the conversion to gray scale of the jpeg image taken by the camera in order to create one matrix filled with the gray scale values representing the image pixels.

Since a jpeg image is composed by three 640x480 matrix components for red, green and blue respectively; is possible to weight these values with some specific constants and create a new matrix filled with gray scale values that are the composition of the original three components.

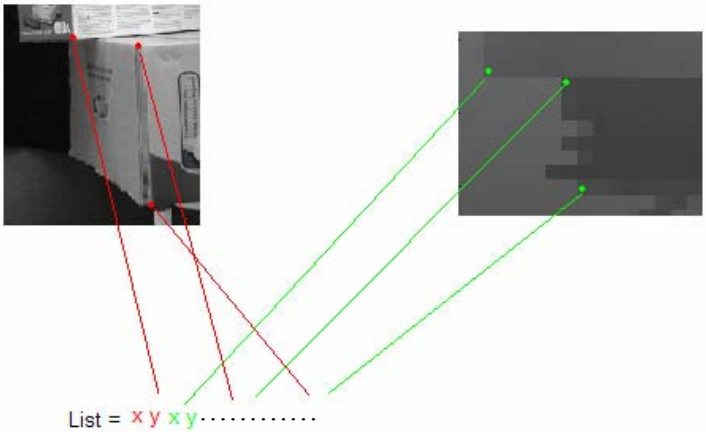
```
for i = 1 : 480
   for j = 1 : 640
      Xrgb(i,j) = 0.2990 * X(i,j,1) + 0.5870 * X(i,j,2) + 0.1140 * X(i,j,3);
      end
end
```

Once the gray scale matrix has been created, is possible to open that image and the z matrix of the scans, we can proceed now with the point matching trying to find the correspondences between the corners manually, that means by clicking with the mouse over them and creating a list with the coordinates of both.



Pict 4 . The gray scale rep. of the image taken by the camera (a) and the image of the Z matrix (b)

As showed in picture 4, all the anchor points in the picture have been highlighted with their correspondent in the data, their coordinates are then matched together and placed in a list like depicted in picture 5.



Picture 5 . Anchor point matching

At this point we have completed the list of anchor points, so is possible to move on applying the formulas.

```
Loop for (Total number of anchor points)

[

Xpic = List (1st)-320 //The X coord. And y coord. of the picture is the first
Ypic = List(2nd)-240    // and the first and second in the group of 4.

Xspace = x(List (3rd), List(4th))  // The x and y and z coordinates of the space
Yspace = y(List (3rd), List(4th))  // are found in their respective matrices at
Zspace = z(List (3rd), List(4th))  // the location give by the 3rd and 4th pos.

A (1st row) = (Xspace, Yspace, Zspace …)  // A matrix is filled with the values
A (2nd row) = (0 0 0 1 …)

]
```

In the piece of code above from the Xpic and Ypic coordinates half the size of the image have been subtracted, that is because we are shifting the starting point in the top-left corner of the image, that means considering 0 for the values u and v in the K matrix as is supposed in its calculation.

After the loop above, the matrix A is then formed. Using the calculator is then possible to find the eigenvector representing our R*K matrix and separate them.

The R matrix will be of size 3x3 and will be used to find the rotation angle φ applying the formula outlined above in the report.

The value of φ obtained is around 23 degrees that can be considered correct since the displacement of the camera was quite small, as is possible to see in the picture 4(a) also.

On the contrary the K matrix obtained, that have to be scaled before in order to get a 1 in the (3,3) position, will be:

$$K = \begin{pmatrix} 17.4 & 0 & 0 \\ 0 & 7.2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Where the first two values in the diagonal represent the focal length of the camera.

Once the rotation matrix has been found is possible to prove the correctness of the algorithm by just multiplying the original data by that matrix, obtaining as a result the rotation in the space of the object.

In figure 10 are depicted both models of the object in the space:

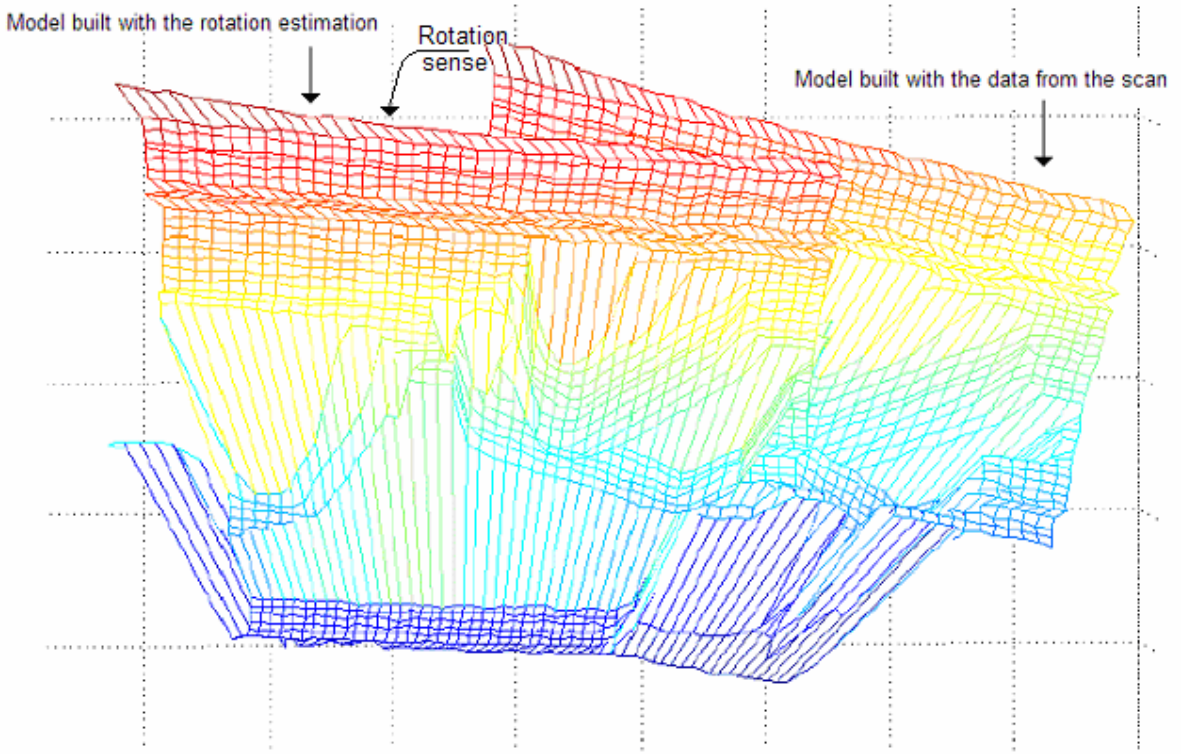
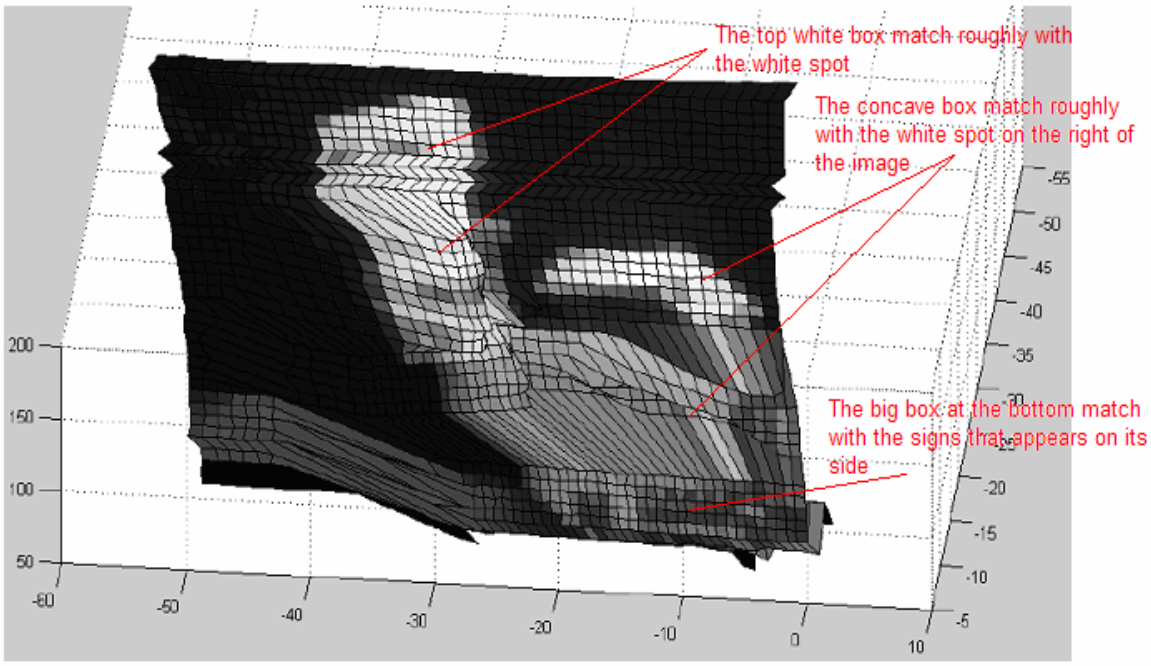


Figure 10 . Representation in the space of the models created with the original data and the rotated one

As is possible to see in figure 10 the two representation are slightly rotated one respect to the other. More exactly the angle between them is roughly the displacement between the camera and the laser, that means that the estimated model has a position respect to the laser that is roughly the same position of the original object respect to the camera.

As a consequence of what we have just explained is the image placing over the model, that means that in order to check our results, is possible to set the picture obtained by the camera previously as the surface of our model and prove that the parts of the image match roughly with the same parts in the model.

The situation obtained is depicted in picture 6:



Picture 6 . Placing the image on the model of the rotated data

The image confirm our theory about the match between the picture and the model, in the figure are highlighted all the matching areas that correspond to the same parts in both representations. The correspondence is not perfect due to many factors, one of them is the solution of the matrix equation to get the rotation matrix; the matrix found with the eigenvectors is close but not really the same as the original one, another factor is the scaling of the picture that have being performed in order to have the size of the image the same as the data and to be able to set it as the texture of the model.

CHAPTER 7

Conclusion and further research

The complete work we have done in this project was formed by different main parts. The first of them was principally the creation of a suitable interface between the user and the device. That means the organization and the design, with a programming language, of the basic functions needed, like all the algorithms in the background of the read and write functions. Especially the synchronization of the flow of the data passing through the serial port, the correct parameters used to set up the port and all the commands needed for a correct read and write action had to be coded.

Another main part was the development of the software, that means finding a good mechanism needed to represent the data received from the packet with a suitable graph, the more similar to the real world that have been scanned. The development of algorithms for the synchronization between the number of the sample and the positions in the graph were required.

The last main part of the project was the nearest to the machine vision world, here is where some principles of that area have been applied. The point matching algorithm and all the mathematical formulation referred to the camera representation and rotation were coded in order to process our data from a machine vision point of view.

In a general view each part of the project have been very useful to learn. In particular in the first coding part I developed my skills to set up a Serial interface with all the important functions embedded. I had the possibility to know about how to use the graphic interface of the programming language when trying to develop the user window and all its working tools.
Finally, and the most important, was the possibility I had in the last part to know more about machine vision. In particular I developed some basic functions for treating images, like conversion to gray scale, scaling the size etc. I could apply also some machine vision principles to my real data and observe the effects and the results obtained. That was a big step I could make in order to go deeper into this interesting science area and to make me know things about its way of working that I would like to increase in the future.

Another important achievement I have done with this project was the knowledge of the Laser Scanner, I had the possibility to know about its way of working under the physical and technical

point of view. The Laser have many possible application fields, I think that the way it works has many potentialities that can be applied for many different purposes. One of them, that I consider to be the must important is related with the position localization, by using its possibilities to retrieve information about the distance of the objects placed in front of it, is possible to get a sort of robotic view, that can be used as sensor to drive in the environment any kind of machine.

As further research, I would think is worth to continue applying the laser as a position localizer. Due to its resolution I wouldn't really consider it suitable for the recreation of 3D models of small objects that wouldn't give brilliant result as those that could be obtained if using the laser for scanning bigger environments.

References

[1] - SICK LMS 200 Data sheet

[2] - "*On Plane-based camera calibration: a general algorithm, singularities, applications*". CVPR- IEEE Conference in computer vision and pattern recognition; Fort, Collins, Colorado Volume 1 pp 432 – 437 June 1999

[3]- McKerrow, P, J. (1991), *Introduction to robotics*, Addison-Wesley.

[4]- Paul P. R. (1981), Robot *manipulators: mathematics, programming and control*, MIT Press