Escola Politècnica Superior
d'Edificació de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

INGENIERÍA TÉCNICA EN TOPOGRAFÍA
PROYECTO FINAL DE CARRERA

MODELADO FOTOGRAMÉTRICO DE OBJETOS MEDIANTE CAPTURA FOTOGRÁFICA DIGITAL Y PLATAFORMA GIRATORIA

Proyectista/es: Daniel Gerardo Sotoca Balsebre

Director/es: Felipe Buill Pozuelo
Albert Prades i Valls

Convocatoria: Marzo 2011

RESUMEN

En este proyecto se han implementado un conjunto de rutinas en lenguaje C++ que sirven para crear un modelo de un pequeño objeto en tres dimensiones. Se parte de un conjunto de fotografías del objeto que contienen información bidimensional y se obtiene su forma tridimensional. En las rutinas se ha tenido en cuenta la corrección de la distorsión de la lente para la eliminación de esta aberración geométrica. Se ha hecho uso de la condición de coplanaridad entre fotogramas consecutivos para la orientación relativa. Por último, se aplica la intersección de haces de luz para obtener el modelo.

Las fotografías han sido tomadas utilizando una cámara no métrica fija sobre un trípode. El objeto se sitúa sobre una plataforma circular giratoria a la que se le han añadido unas marcas en el perímetro para controlar el giro. El modelo final se presenta en formato *DXF* y se puede visualizar con cualquier programa de diseño gráfico compatible.

Este trabajo pertenece a una colección de PFC'S que se han realizado en el laboratorio de fotogrametría y puede ser utilizado como base para realizar futuros proyectos aprovechando el código creado.

ÍNDICE

1 INTRODUCCIÓN 5

2 PROGRAMA “MODEL” 7

2.1 TOMA DE DATOS 7

2.1.1 OBTENCIÓN DEL BLOQUE FOTOGRÁFICO 7

2.1.2 EXTRACCIÓN DE LA INFORMACIÓN DIGITAL 9

2.2 ORIENTACIÓN RELATIVA 15

2.2.1 PARÁMETROS INICIALES DE CÁMARA 15

2.2.2 CONDICIÓN DE COPLANARIDAD 16

2.2.3 MONTANJE DEL SISTEMA DE ECUACIONES 18

2.2.4 COORDENADAS MODELO 20

2.2.5 MODELO FINAL 22

3 CONCLUSIONES 23

4 BIBLIOGRAFIA 24

5 AGRADECIMIENTOS 25

1 INTRODUCCIÓN

Existe instrumental que permite realizar una captura masiva de datos de objetos y otros elementos. Tenemos algunos ejemplos en el LaserScan y el sónar que se utiliza para realizar batimetrías. El objetivo de este proyecto es crear un programa que permita obtener la mayor cantidad de datos con el menor coste posible y de un modo original.

La particularidad de este proyecto viene dada por la característica principal que la cámara utilizada se encuentra fija en un punto y es el objeto el que situado sobre una plataforma giratoria se mueve mientras es fotografiado cada pocos grados de giro.

En la memoria se detallan los procesos seguidos para la creación del programa, las rutinas que se han desarrollado y las clases que se han usado para apoyar este proyecto.

2 PROGRAMA “MODEL”

2.1 TOMA DE DATOS

2.1.1 OBTENCIÓN DEL BLOQUE FOTOGRÁFICO

Antes de empezar a programar se necesitan algunas fotografías para poder trabajar en programa de pruebas. Tomamos como sujeto de pruebas una pequeña estatua con unas dimensiones aproximadas de 10x8x4cm con forma de tótem. Vemos el objeto en la figura 2.1.1.b.
Para la toma del bloque fotográfico se han utilizado los siguientes elementos:

- Peana giratoria de plástico de 30 cm de diámetro con una cinta reglada en el perímetro.
- Cámara de fotos CANON modelo EOS 450D.
- Trípode BENRO modelo a-350EX con rótula BH0-M.

Figura 2.1.1: Toma de datos.

Se debe tener en cuenta que las fotografías han de tomarse desde un punto un poco más elevado que el plano horizontal donde se encuentra el objeto para poder reducir al máximo los reflejos.

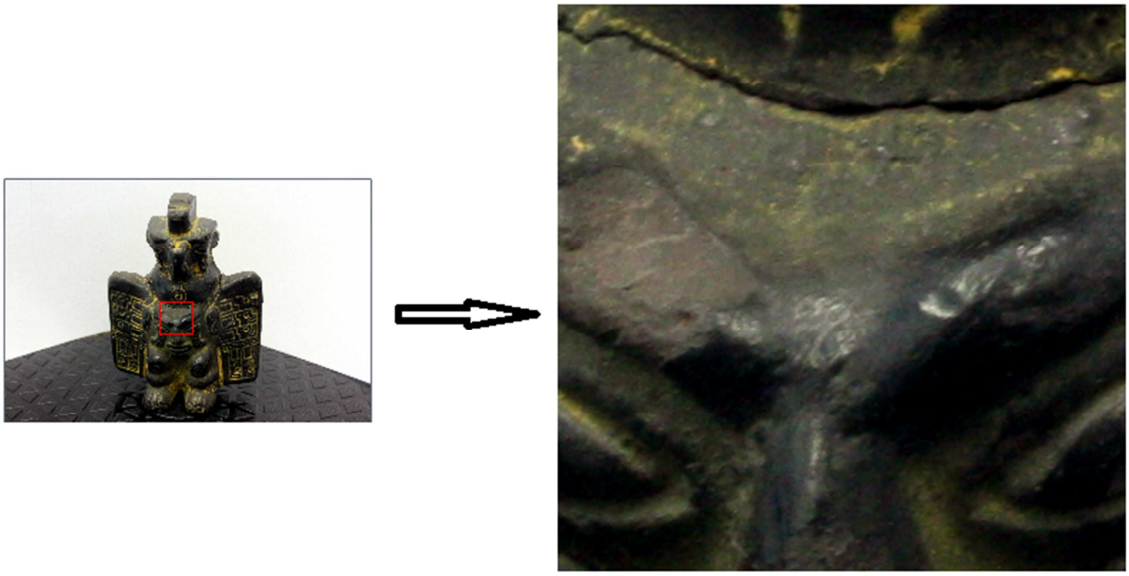


Figura 2.1.1: Muestra de zonas con reflejos en la figura utilizada.

Una vez se ha realizado la toma de datos se procede a la extracción de la información digital.

2.1.2. EXTRACCIÓN DE LA INFORMACIÓN

Las fotografías tienen un tamaño medio de 4 MB y unas dimensiones de 4272 columnas y 2848 filas en formato JPEG, un formato de fotografía con el que no podemos trabajar de un modo adecuado. Decidimos por tanto variarlo.
Las fotografías se convierten a imágenes tipo *.IMG*. Éste formato guarda la información digital en un mapa de bits donde cada píxel contiene la información de los tres canales RGB (Red, Green, Blue). En la figura 2.1.2.a podemos ver la imagen en cada canal.

Figura 2.1.2.a: Tres canales de la misma imagen.

Este formato hace que cada canal tenga un tamaño de unos 12 MB, haciendo que cada fotografía tenga un tamaño aproximado de 35 MB. Debido al tamaño de las imágenes y a sus dimensiones se toma la decisión de trabajar con un solo canal.
Para hacer eso variamos el modo en que se representan las imágenes haciendo que pasen de estar representadas mediante la codificación RGB a la codificación HSI (hue, saturation & intensity), utilizando únicamente el canal de la intensidad en cada fotografía en escala de grises. En la figura 2.1.2.b podemos ver un ejemplo.

Figura 2.1.2.b: Canal de intensidad en escala de grises.

Si se utiliza éste método, el tamaño es el de un solo canal, haciendo mucho más manejables las imágenes sin perder una gran cantidad de información.
Las dimensiones de las fotografías también presentan problemas. Las dimensiones que tienen las imágenes hace que el tiempo de cada rutina sea muy elevado, haciéndonos tomar en cuenta el método de la creación de pirámides de imágenes para poder trabajar con fotogramas mucho más pequeños y de este modo que el tiempo se reduzca.

PIRÁMIDE DE IMÁGENES

Las pirámides de imágenes son la reducción del tamaño de una imagen. Normalmente se elige reducirla a la mitad, tanto horizontal como verticalmente. Esto hace que cada nuevo nivel de la pirámide mida una cuarta parte respecto al anterior.
El proceso calcula la media de un conjunto de pixeles y aplica ese valor al pixel en la posición correspondiente en la nueva imagen.
Veamos un ejemplo numérico en la tabla 2.1.2.c:

Tabla 2.1.2.c: Pirámide de imágenes

Imágen 1

45	43	45	47
48	41	42	43
49	44	45	44
51	47	46	45

Imágen 2

44	44
48	45

Este método se repite hasta que se obtienen imágenes menores a un determinado tamaño. En el caso del programa, realizará pirámides hasta que haya reducido el tamaño de la imagen a menos de 1000 columnas y 500 filas. Se ha elegido este tamaño de imagen por ser suficientemente grande como para poder observar la imagen con algún detalle y ser suficientemente pequeña para que el programa pueda analizarla completamente en pocos segundos.
En la figura 2.1.2.d que tenemos a continuación se pueden observar los tamaños en las tres reducciones realizadas con el programa y en la figura 2.1.2.e: la pérdida de detalle que se

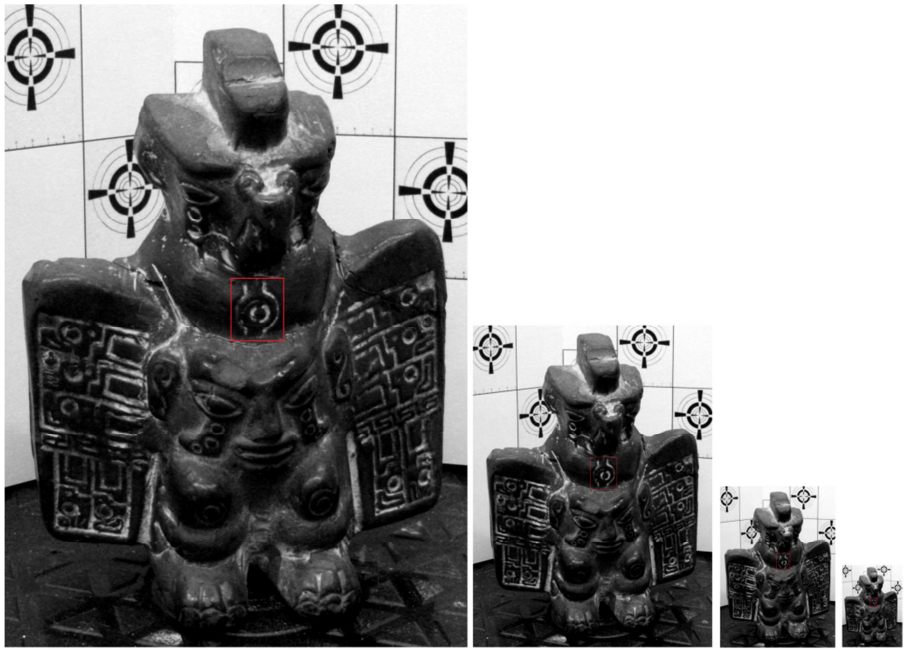


Figura 2.1.2.d: Diferencia de tamaños en las pirámides de imágenes.

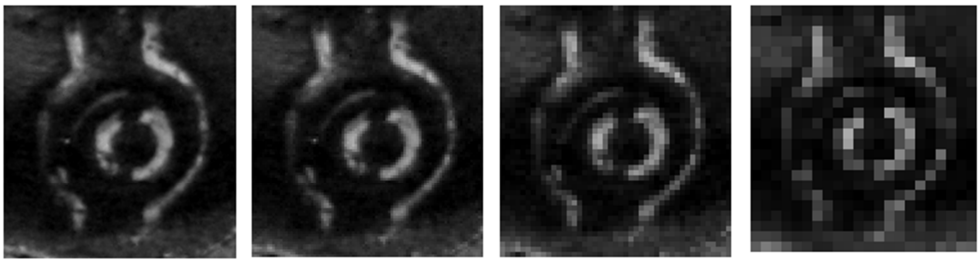


Figura 2.1.2.e: Comparación del detalle en las imágenes.

Código de programación para la realización de las pirámides de imágenes:

```
void CFotograma::PiramideDeImatges(CString nom)
{

register UINT i,j;
CFile fitxer;
BYTE *pBYTEima;
BYTE *pBYTEaux;

  for( UINT k=0 ; k<m_nNivells-1 ; k++ )
  {
   //Rutina para abrir imágenes
    pBYTEima= new BYTE[3*m_nColumnesPiramide[k]*m_nFilesPiramide[k]];
    fitxer.Open( m_strNoms[k], CFile::modeRead | CFile::typeBinary  );
    fitxer.Read( pBYTEima , 3*m_nColumnesPiramide[k]*m_nFilesPiramide[k] );
    fitxer.Close();

    pBYTEaux= new BYTE[3*m_nColumnesPiramide[k+1]*m_nFilesPiramide[k+1]];

   //Crear la pirámide a partir de la imagen anterior.
    for( j=0 ; j<m_nFilesPiramide[k+1] ; j++)
      for( i=0 ; i<3*m_nColumnesPiramide[k+1]-3 ; i+=3)
      {
        pBYTEaux[3*m_nColumnesPiramide[k+1]*j+i]=
                (BYTE)((pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(i*2)]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(2*i+3)]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2)]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2+3)])/4.0);
        pBYTEaux[3*m_nColumnesPiramide[k+1]*j+i+1]=
                (BYTE)((pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(i*2)+1]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(i*2+3)+1]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2)+1]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2+3)+1])/4.0);
        pBYTEaux[3*m_nColumnesPiramide[k+1]*j+i+2]=
                (BYTE)((pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(i*2)+2]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2)+(i*2+3)+2]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2)+2]
                +pBYTEima[3*m_nColumnesPiramide[k]*(j*2+1)+(i*2+3)+2])/4.0);
      }

   //Almacenar la nueva pirámide como una imagen.
    fitxer.Open( m_strNoms[k+1], CFile::modeCreate | CFile::modeWrite | CFile::typeBinary  );
    fitxer.Write( pBYTEaux ,  3*m_nColumnesPiramide[k+1]*m_nFilesPiramide[k+1] );
    fitxer.Close();

    delete [] pBYTEima;
    delete [] pBYTEaux;
  }
```

El tamaño de las imágenes más pequeñas hace que el programa opere mucho más rápido. Una vez se han terminado de hacer se procede a la búsqueda de puntos de interés en las imágenes.

LOS PUNTOS DE INTERÉS

Los puntos de interés son puntos de la imagen con un valor radiométrico muy diferente al valor que tienen el resto de puntos de alrededor. Un píxel blanco sobre fondo negro es, por ejemplo, un píxel de interés ya que su valor digital es muy distinto al del resto de alrededor, haciéndolo fácilmente identificable de resto.
Ésta rutina utiliza el método descrito en [Moravec] consistente en calcular las desviaciones tipo de cada punto en las cuatro direcciones principales. En nuestro caso para ahorrarnos cálculo intensivo de raíces cuadradas lo calculamos con la varianza. Da a ese punto la menor de las cuatro calculadas y éste cálculo se hace para cada píxel de la imagen. Entonces, sobre este resultado se buscan los máximos locales. En la figura 2.1.2.d se puede ver el cálculo de las varianzas para un punto.

$$s_{min} = min\{s_1, s_2, s_3, s_4\}$$

$$s_1 = \frac{1}{N^2}\sum_{x=1}^{N}\sum_{y=1}^{N}(I(x,y) - I(x+1,y))^2$$

$$s_2 = \frac{1}{N^2}\sum_{x=1}^{N}\sum_{y=1}^{N}(I(x,y) - I(x,y+1))^2$$

$$s_3 = \frac{1}{N^2}\sum_{x=1}^{N}\sum_{y=1}^{N}(I(x,y) - I(x+1,y+1))^2$$

$$s_4 = \frac{1}{N^2}\sum_{x=1}^{N}\sum_{y=1}^{N}(I(x,y) - I(x+1,y-1))^2$$

Figura 2.1.2.d: Algoritmo de Moravec.

Se trata de un método utilizado para poder reducir la posterior búsqueda de puntos homólogos descartando los que no tengan una fácil identificación. De este modo, nos centramos solamente en un pequeño grupo de puntos seleccionados descartando al resto, que solamente harían perder tiempo al programa.
En la figura 2.1.2.e se pueden ver los puntos seleccionados por este método. Como se observa, sobre el fondo blanco no se detecta ningún punto y por lo tanto se pasará por alto evitando hacer perder el tiempo y los recursos del programa.

Figura 2.1.2.e: Puntos de interés.

Los puntos de interés encontrados son almacenados en un fichero ASCII que nos da las coordenadas imagen de cada punto de interés del fotograma.
Este archivo es el que se utilizará posteriormente para la búsqueda de puntos homólogos.

LOS PUNTOS HOMÓLOGOS: LA CORRELACIÓN AUTOMÁTICA

Cuando hemos encontrado los puntos de interés de una de las imágenes necesitamos encontrar esos puntos en el fotograma consecutivo. Para ello utilizaremos la función de correlación.
Partiendo de un punto de interés creamos un patrón centrado en él (en nuestro caso se trata de matrices cuadradas de 25 píxeles de lado). En el segundo fotograma establecemos una zona de búsqueda un poco mayor que el patrón (de 55 píxeles en horizontal y 30 en vertical) sobre la que se va a calcular la función de correlación.

$$p(j,k) = \frac{\sum_{x=1}^{N}\sum_{y=1}^{M}\left[(I_c(x,y)-\bar{I}_c)(I_p(x,y)-\bar{I}_p)\right]}{\sqrt{\sum_{x=1}^{N}\sum_{y=1}^{M}(I_c(x,y)-\bar{I}_c)^2}\sqrt{\sum_{x=1}^{N}\sum_{y=1}^{M}(I_p(x,y)-\bar{I}_p)^2}}$$

Figura 2.1.2.f: Correlación.

Valor radiométrico del patrón de referencia: I_p
Valor radiométrico del patrón de búsqueda: I_c

Una vez calculadas todas las correlaciones en la zona de búsqueda encontraremos el máximo local.
La función de correlación requiere diversos cálculos que detallo a continuación:
En primer lugar se calcula el valor medio de los niveles digitales del patrón y la desviación tipo.
En la zona de búsqueda del segundo fotograma se hacen recortes del mismo tamaño que el patrón de búsqueda y se calcula para esas porciones el valor medio, la desviación estándar y la covarianza con el patrón. De este modo se puede calcular la correlación para todas las posiciones de la matriz de búsqueda. Una vez calculadas se busca el máximo local y almacenamos su posición en la segunda imagen.
El valor de la función de correlación es un valor acotado entre -1 y 1, siendo el 1 una muestra exactamente igual de los patrones y el -1 el negativo del patrón de referencia. En la figura 2.1.2.g se observan ambos casos.

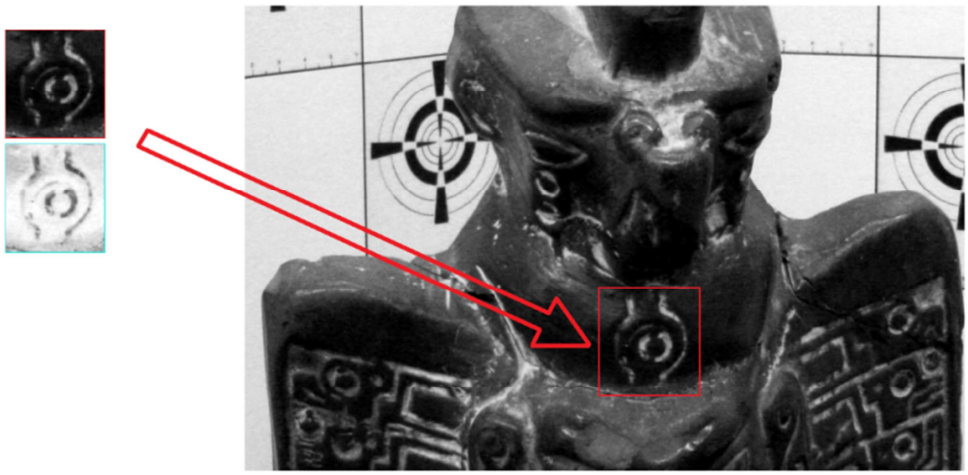


Figura 2.1.2.g: Muestra de dos patrones extremos.

Cabe destacar que este no es un proceso perfecto. Una correlación baja o negativa (supongamos entre el 0,7 y el -1) nos da puntos que no son similares en la mayoría de los casos, no en todos, mientras que una covarianza alta (0,7 a 0.99), pese a que se trata de valores elevados no nos da garantías de que haya encontrado los puntos homólogos correctamente.
La correlación se calcula primero en los puntos de interés de la pirámide más pequeña. Por tanto es necesario tener un margen de búsqueda en ese nivel. Si pasa la tolerancia, se realiza una búsqueda mucho más concentrada en la pirámide siguiente tomando como referencia las posiciones anteriores con un margen de 2 o 3 píxeles para dar más posibilidades a encontrar el punto correcto. Contrariamente a lo acontecido en la primera pirámide (si la tolerancia era válida inmediatamente se calcula en el siguiente nivel de la pirámide), a partir de aquí el programa elige la correlación más alta en todos los niveles hasta llegar al último. Si también pasa el último nivel se calcula la corrección radial para almacenar las fotocoordenadas, se calculan unas coordenadas modelo aproximadas y se almacena todo.
En un estudio donde se comprueban las covarianzas de 20 puntos de interés de un fotograma se han encontrado un 75% de puntos satisfactoriamente, un 15% de puntos estaban muy próximos entre sí y el 10% eran malas correlaciones.
Para mejorar la precisión original de esta búsqueda, originalmente de un píxel de precisión, interpolamos los valores de las correlaciones próximas. La interpolación horizontal se utiliza en el caso de la coordenada *x* y para la coordenada *y* se usa la vertical. En la figura 2.1.2.h podemos ver un esquema de este cálculo:

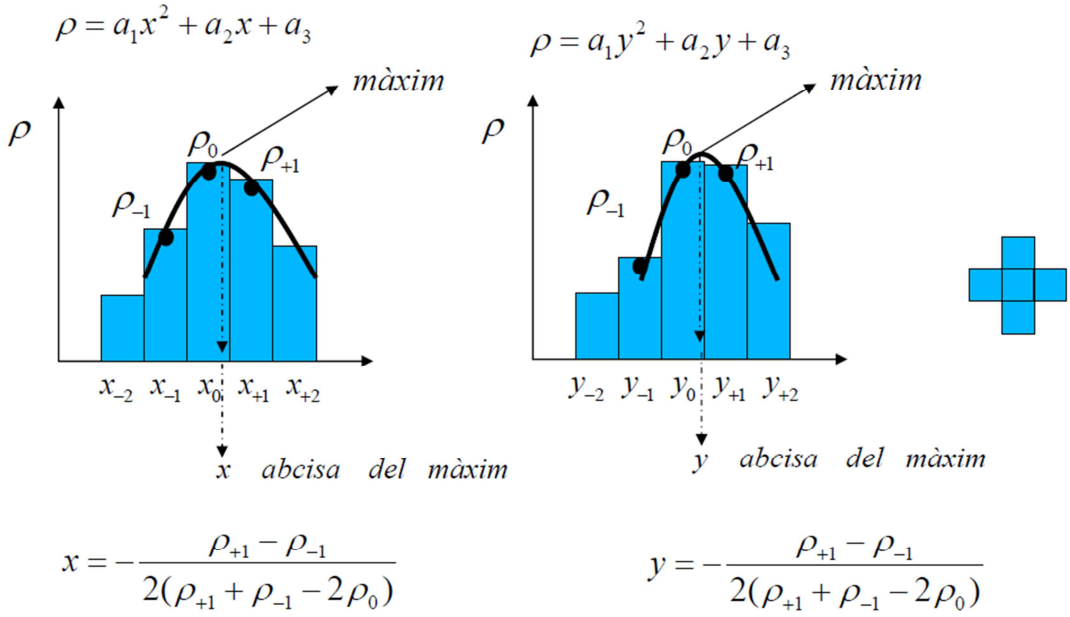


Figura 2.1.2.h: Interpolación.

Una vez se ha terminado la búsqueda de puntos homólogos ya podemos dar por terminada la toma de datos y pasamos al cálculo del sistema.

2.2 ORIENTACIÓN RELATIVA:

Para la resolución del sistema se opta por realizar una orientación relativa mediante el uso de la condición de coplanaridad.

2.2.1 PARÁMETROS INICIALES DE CÁMARA

El primer paso para realizar la orientación relativa del sistema es obtener unos parámetros iniciales de cámara.
En este problema solamente existe una posición de la cámara que se encuentra fija sobre un trípode y es el objeto el que se mueve. No obstante, optaremos por establecer el modelo exactamente al revés, suponiendo que es el objeto el que se mantiene estático y es la cámara la que se mueve alrededor de él. En la figura 2.2.1 se puede observar un pequeño esquema para dar una idea del modelo.

Figura 2.2.1: Ejemplo del planteamiento del sistema de cámaras.

Para poder calcular las posiciones en el espacio y los ángulos de la cámara en el momento de las tomas es necesario calcularlos *a priori* de un modo aproximado, encontrando los valores más precisos mediante el posterior ajuste mínimo cuadrático.
Giros de cámara:

$$\omega_i = 5°$$
$$\varphi_i = 3{,}5°$$
$$\kappa = 0°$$

De los tres ángulos ω, φ y κ solamente se modifican ω y φ, siendo ω constante ya que el giro se debe a una inclinación constante de la cámara. El ángulo φ se da *a priori* ya que las marcas de la peana circular tienen una distancia angular aproximada que ya conocemos.

Cálculo de las posiciones del centro de proyección de la cámara:

$$X_0^i = X_0^1 + Z_0^1 \cos\left(\frac{\pi}{2} - \varphi_i\right)$$
$$Y_0^i = Y_0^1$$
$$Z_0^i = Z_0^1 \ \operatorname{sen}\left(\frac{\pi}{2} - \varphi_i\right)$$

Se considera i como el número de fotografías del bloque.

La componente *Y* permanece constante ya que se puede asumir que el plano de giro es totalmente perpendicular a la vertical del sistema.
Como se puede observar es bastante sencillo para este sistema crear unos valores aproximados *a priori*, haciendo mucho más simple este apartado de cálculo.

2.2.2 CONDICIÓN DE COPLANARIDAD

Se impone que los rayos que pertenecen a los puntos homólogos se corten y formen un plano, cuya ecuación es:

$$aX + bY + cZ + d = 0$$

Como los dos centros de proyección y las dos imágenes del punto forman un plano tenemos que se debe cumplir la condición siguiente:

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 \\ X_3 & Y_3 & Z_3 & 1 \\ X_4 & Y_4 & Z_4 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

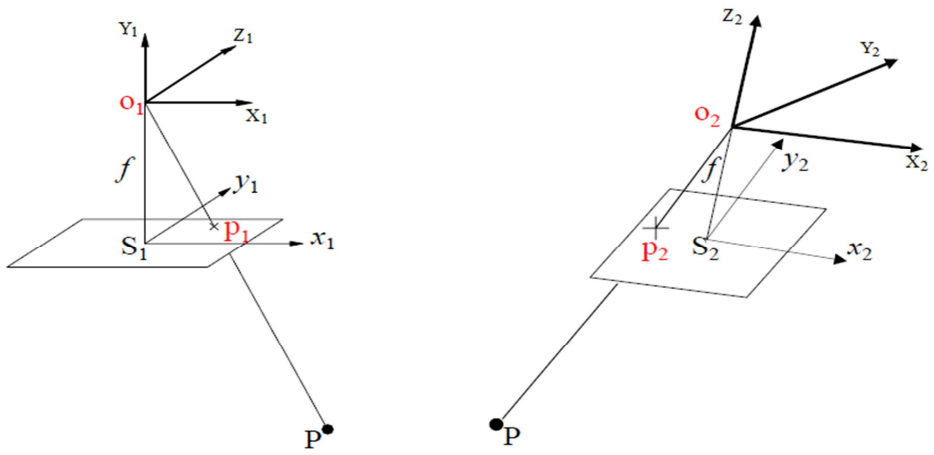
Para que los cuatro puntos sean coplanarios el determinante debe ser igual a 0:

$$\begin{vmatrix} X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 \\ X_3 & Y_3 & Z_3 & 1 \\ X_4 & Y_4 & Z_4 & 1 \end{vmatrix} = 0$$

Siendo en este caso las definiciones siguientes de los centros de proyección de ambos fotogramas y de las imágenes de los puntos:

$$O_1 \rightarrow (X_3 \quad Y_3 \quad Z_3)$$
$$O_2 \rightarrow (X_4 \quad Y_4 \quad Z_4)$$
$$P_1 \rightarrow (X_1 \quad Y_1 \quad Z_1)$$
$$P_2 \rightarrow (X_2 \quad Y_2 \quad Z_2)$$

Como podemos ver por las ecuaciones se deben encontrar doce parámetros en total, seis pertenecen a los giros de los fotogramas y los seis restantes a las coordenadas de los centros de la imagen como se puede ver en la imagen 2.2.2.a



$\Delta\kappa_1,\ \Delta\omega_1,\ \Delta\phi_1$
$\Delta x_1,\ \Delta y_1,\ \Delta z_1$

$\Delta\kappa_2,\ \Delta\omega_2,\ \Delta\phi_2$
$\Delta x_2,\ \Delta y_2,\ \Delta z_2$

Figura 2.2.1: Esquema de la orientación relativa.

Si fijamos el primer fotograma y la coordenada *X* del segundo fotograma es posible crear un sistema donde únicamente hay cinco incógnitas que permiten resolver el sistema del siguiente modo:

$$\begin{vmatrix} 1 & b_y & b_z \\ x_1' & y_1' & z_1' \\ x_2' & y_2' & z_2' \end{vmatrix} = 0$$

Donde b_y y b_z son las bases de los ejes *Y* y *Z* respectivamente y el resto son las coordenadas de los puntos homólogos.
Refiriendo las coordenadas del punto dos al primer fotograma obtenemos:

$$\begin{pmatrix} x_1' \\ y_1' \\ z_1' \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ -f \end{pmatrix} \qquad \begin{pmatrix} x_2' \\ y_2' \\ z_2' \end{pmatrix} = \begin{pmatrix} U_2 \\ V_2 \\ W_2 \end{pmatrix} - \begin{pmatrix} 1 \\ b_y \\ b_z \end{pmatrix} = \mathbf{R}\begin{pmatrix} x_2 \\ y_2 \\ -f \end{pmatrix} - \begin{pmatrix} 1 \\ b_y \\ b_z \end{pmatrix}$$

Siendo *R* la matriz de rotación en el espacio obtenemos que:

$$R = \begin{pmatrix} \cos\kappa\ \cos\varphi & \operatorname{sen}\kappa\cos\varphi & -\operatorname{sen}\varphi \\ \operatorname{sen}\varphi\ \operatorname{sen}\omega\ \cos\kappa - \operatorname{sen}\kappa\ \cos\omega & \operatorname{sen}\kappa\ \operatorname{sen}\omega\ \operatorname{sen}\varphi + \cos\kappa\ \cos\omega & \cos\varphi\ \operatorname{sen}\omega \\ \operatorname{sen}\varphi\ \cos\omega\ \cos\kappa + \operatorname{sen}\omega\ \operatorname{sen}\kappa & \operatorname{sen}\varphi\ \cos\omega\ \operatorname{sen}\kappa - \cos\kappa\ \operatorname{sen}\omega & \cos\varphi\ \cos\omega \end{pmatrix}$$

$$\begin{pmatrix} U_2 \\ V_2 \\ W_2 \end{pmatrix} = \mathbf{R}\begin{pmatrix} x_2 \\ y_2 \\ -f \end{pmatrix}$$

Donde la ecuación de coplanaridad finalmente queda del siguiente modo:

$$\Delta = \begin{vmatrix} 1 & b_y & b_z \\ x_1 & y_1 & -f \\ U_2 & V_2 & Z_2 \end{vmatrix} = 0$$

Desarrollada es:

$$\Delta = \left(f\ b_y + b_z\ y_1 \right) U_2 - \left(f\ b_z\ x_1 - f \right) V_2 - \left(y_1 - b_y\ x_1 \right) W_2 = 0$$

Como podemos ver se trata de una ecuación no lineal. Luego hay que linealizarla. Para ello hacemos un desarrollo en serie de Taylor donde nos quedamos con los términos de primer orden y despreciamos los de orden superior.

$$v_i = d\Delta_i + \Delta_i = \left(\frac{\partial \Delta}{\partial b_y}\right) db_y + \left(\frac{\partial \Delta}{\partial b_z}\right) db_z + \left(\frac{\partial \Delta}{\partial \omega}\right) d\omega + \left(\frac{\partial \Delta}{\partial \varphi}\right) d\varphi + \left(\frac{\partial \Delta}{\partial \kappa}\right) d\kappa + \Delta_i$$

Una vez linealizada la ecuación para un punto queda:

$$\begin{aligned} v_i = {} & (-x_1\ W_2 - f\ U_2\ ) db_y + (-x_1\ V_2 - y_1 U_2\ ) db_z + (MV_2 - NW_2) d\omega + \\ & + [-MU_2 \cos\omega + NU_2 \operatorname{sen}\omega - Q(W_2 \cos\omega - V_2 \operatorname{sen}\omega] d\varphi + \\ & + [M(m_{32}x_2 - m_{31}y_2) + N(m_{22}x_2 - m_{21}y_2) d\varphi - Q(m_{12}x_2 - m_{11}y_2)] d\kappa + \\ & + MW_2 + NV_2 - QU_2 \end{aligned}$$

Siendo:

$$\begin{aligned} M &= y_1 - b_y x_1 \\ N &= f + b_z x_1 \\ Q &= f b_y + b_z y_1 \end{aligned}$$

2.2.3 MONTAJE DEL SISTEMA DE ECUACIONES

Una vez tenemos las ecuaciones linealizadas podemos resolver el sistema. Como hemos visto trabajamos con un par de fotogramas. Si se obtienen suficientes fotocoordenadas de ambos es posible calcular los cinco parámetros del segundo fotograma. Los parámetros a calcular son los siguientes:

1- Ángulos relativos entre los dos fotogramas: ω , φ , κ
2- Coordenadas relativas de los centros de proyección de un fotograma respecto del otro: Y , Z

Es necesario establecer unos valores iniciales para los parámetros que se quieren estimar. En nuestro caso hemos definido los giros entre cámaras como nulos ($\omega = \varphi = \kappa = 0$) y la componente *x* de la base como la unidad ($b_y = b_z = 0$).
Como hemos podido ver, contamos con 5 incógnitas pero el número de ecuaciones del sistema es mucho mayor, por tanto nos encontramos con un sistema sobredeterminado que nos permite resolver la orientación por un método de ajuste mínimo cuadrático.
Planteamos el sistema:

$$A\hat{x} = U$$

Dónde:

Matriz de diseño: A
Vector de términos independientes: U
Vector diferencial de parámetros: $\hat{x}$

Si resolvemos el sistema quedaría del siguiente modo. La matriz de pesos optamos por descartarla ya que ninguna medida se ha realizado de un modo diferente.

$$\hat{x} = (A^t\ A)^{-1}\ (A^t\ U)$$

$$\hat{x} = N^{-1}\ L$$

Encontrando los diferenciales de los parámetros iniciales establecidos.
La matriz de diseño se crea del siguiente modo:

$$A = \begin{pmatrix} \frac{\partial \Delta_1}{\partial b_y} & \frac{\partial \Delta_1}{\partial b_z} & \frac{\partial \Delta_1}{\partial \omega} & \frac{\partial \Delta_1}{\partial \varphi} & \frac{\partial \Delta_1}{\partial \kappa} \\ & & \vdots & & \\ \frac{\partial \Delta_N}{\partial b_y} & \frac{\partial \Delta_N}{\partial b_z} & \frac{\partial \Delta_N}{\partial \omega} & \frac{\partial \Delta_N}{\partial \varphi} & \frac{\partial \Delta_N}{\partial \kappa} \end{pmatrix}_0$$

Siendo sus dimensiones iguales al número de N puntos medidos en el caso de las filas y el número de incógnitas, cinco, en el de las columnas.

Aquí podemos ver los vectores de términos independientes y diferencial de parámetros:

$$\hat{x} = \begin{pmatrix} db_y \\ db_z \\ d\omega \\ d\varphi \\ d\kappa \end{pmatrix} \qquad U = \begin{pmatrix} MW_2 + NV_2 - QU_2|^1 \\ \vdots \\ MW_2 + NV_2 - QU_2|^N \end{pmatrix}_0$$

Las dimensiones del vector de términos independientes son iguales al número de columnas de la matriz de diseño y las del vector diferencial de parámetros al de las columnas.

Para resolver el sistema es necesario disponer de un número mayor o igual de 5 puntos (o ecuaciones); en nuestro caso, al ser mayor, obtenemos un sistema sobre determinado. En la figura 2.2.3.b podemos ver un esquema del proceso mínimo cuadrático.

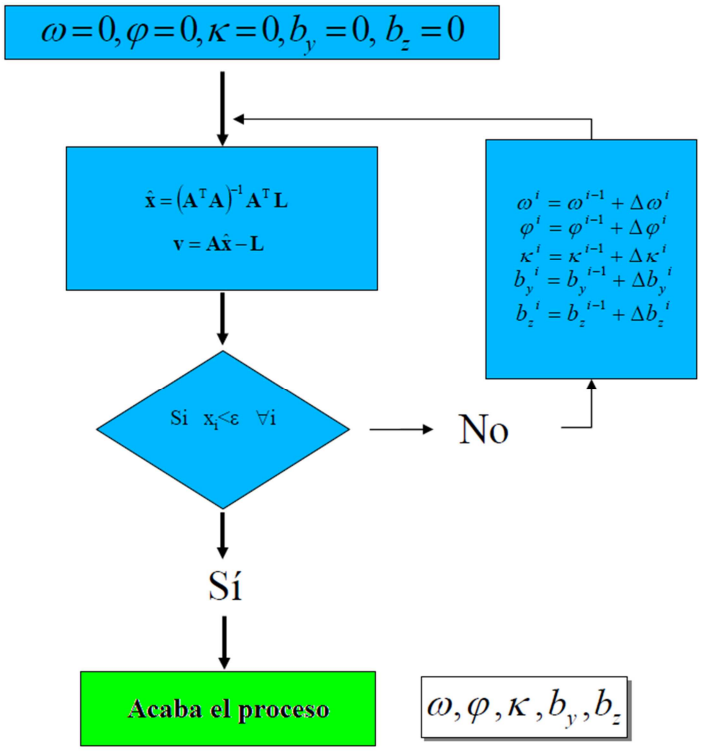


Figura 2.2.3.b: Diagrama de flujo del proceso.

Cuando termina el proceso iterativo obtenemos los parámetros de los fotogramas necesarios para calcular las coordenadas modelo finales que se utilizarán para constituir el modelo tridimensional.

2.2.4 COORDENADAS MODELO

Para la obtención de las coordenadas se utilizan las ecuaciones de [Lerma] que mostramos a continuación:

La matriz de rotación queda del siguiente modo:

$$R = \begin{pmatrix} \cos\kappa\ \cos\varphi & \operatorname{sen}\kappa\cos\varphi & -\operatorname{sen}\varphi \\ \operatorname{sen}\varphi\ \operatorname{sen}\omega\ \cos\kappa - \operatorname{sen}\kappa\ \cos\omega & \operatorname{sen}\kappa\ \operatorname{sen}\omega\ \operatorname{sen}\varphi + \cos\kappa\ \cos\omega & \cos\varphi\ \operatorname{sen}\omega \\ \operatorname{sen}\varphi\ \cos\omega\ \cos\kappa + \operatorname{sen}\omega\ \operatorname{sen}\kappa & \operatorname{sen}\varphi\ \cos\omega\ \operatorname{sen}\kappa - \cos\kappa\ \operatorname{sen}\omega & \cos\varphi\ \cos\omega \end{pmatrix}$$

$$R = \begin{pmatrix} \mathrm{m}_{11} & \mathrm{m}_{12} & m_{13} \\ \mathrm{m}_{21} & \mathrm{m}_{22} & \mathrm{m}_{23} \\ \mathrm{m}_{31} & \mathrm{m}_{32} & \mathrm{m}_{33} \end{pmatrix}$$

Para los valores iniciales de las coordenadas modelo tomamos las fotocoordenadas del primer fotograma y la focal para la profundidad. np corresponde al número de puntos.

$$X_{np}^1 = x_{np}^1$$
$$Y_{np}^1 = y_{np}^1$$
$$Z_{np}^1 = -f$$

U, V, y W son las coordenadas del punto observado en el segundo fotograma referidas al primer fotograma.

$$U = m_{11}\, x_{np}^2 + m_{12}\, y_{np}^2 - f\, m_{13}$$
$$V = m_{21}\, x_{np}^2 + m_{22}\, y_{np}^2 - f\, m_{23}$$
$$W = m_{31}\, x_{np}^2 + m_{32}\, y_{np}^2 - f\, m_{33}$$

Los parámetros de las rectas λ, μ se pueden interpretar como las escalas de los dos fotogramas:

$$\mu = \frac{b_X\, Z_{np}^1 - b_z\, X_{np}^1}{x_{np}^1\, W - Z_{np}^1\, U}$$
$$\lambda = \frac{b_X\, W - b_z\, U}{x_{np}^1\, W - Z_{np}^1\, U}$$

Y por último se obtienen las coordenadas modelo:

$$X_{model} = \lambda\, X_{np}^1$$
$$Y_{model} = \frac{1}{2}\left(\lambda\, Y_{np}^1 + \mu\, V + b_Y\right)$$
$$Z_{model} = \lambda\, Z_{np}^1$$

Es importante que las fotocoordenadas estén bien medidas ya que esto permitirá una mayor aproximación a las coordenadas modelo finales. Se puede determinar si una coordenada es fiable o no observando la paralaje vertical a la hora de calcular la coordenada Y_{model}. Si la diferencia entre las componentes (sin tener en cuenta la base) es elevada, ese punto se considera malo y se desprecia. La diferencia entre un modelo final con coordenadas que

tienen mucha o poca paralaje es significativa. En la figura 2.2.4 podemos ver un ejemplo del mismo cálculo añadiendo o no el filtro de la paralaje:

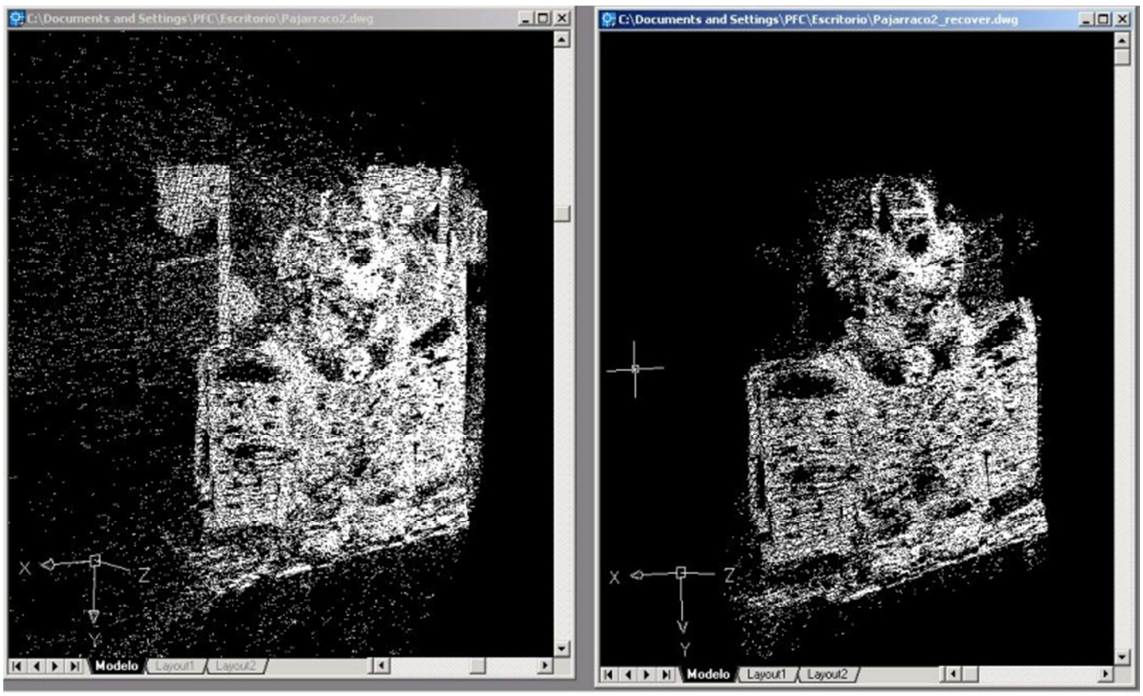


Figura 2.2.4: modelo sin filtrar y modelo filtrado con paralaje vertical.

Las coordenadas resultantes se almacenan en un archivo ASCII con extensión .txt. Con ese archivo es posible crear el modelo final.

2.2.5 MODELO FINAL

Cuando se ha terminado todo el proceso el programa da el tiempo total de cálculo y el número de puntos que han pasado la correlación y sus coordenadas modelo. Éstas se pueden importar a un archivo .dxf con AutoCAD® y ver el resultado final. En la figura 2.2.5.a podemos ver una captura de pantalla del programa al finalizar el cálculo y en la figura 2.2.5.b podemos ver el modelo final y una comparación con la fotografía origen.

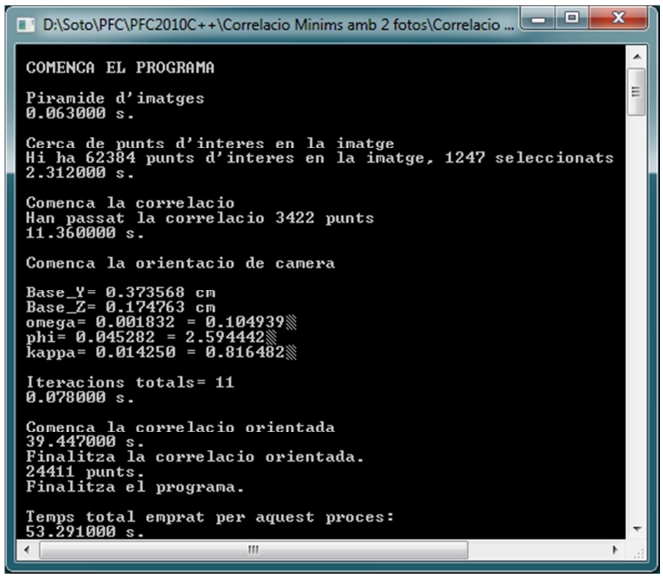


Figura 2.2.5: Final del cálculo.

Modelo final comparado con la figura inicial.

Figura 2.2.5: Modelo final.

3 CONCLUSIONES

1) La primera fase del proyecto, de aproximadamente un mes de duración, se invirtió en el aprendizaje de programación en lenguaje C++.

2) El programa final está dividido en funciones que resuelven los siguientes puntos:

   - Creación automática de las pirámides de imágenes en tres niveles utilizando el canal intensidad.
   - Obtención de los puntos de interés de las imágenes.
   - Búsqueda de los puntos homólogos entre dos fotogramas consecutivos.
   - Interpolación de las coordenadas imagen.
   - Corrección de la distorsión de la lente de las fotocoordenadas medidas.
   - Cálculo de la orientación relativa entre dos fotogramas consecutivos valiéndose de la condición de coplanaridad.
   - Cálculo de las coordenadas modelo por la intersección de haces de luz en el espacio.

3) Este trabajo se ha realizado en el seno del laboratorio de fotogrametría y forma parte de uno de los proyectos de R+D+I que se realizan en él.

4 BIBLIOGRAFÍA

Lerma, J.L. (2002), Fotogrametría moderna: analítica y digital. Editorial: Universidad Politécnica de Valencia, 2002.

Moravec , H. P. (1977) Towards Automatic Visual Obstacle Avoidance. Proceedings of the 5th International Joint Conference on Artificial Intelligence, MIT, Cambridge, Mass.

Press, W.H. et al. (2002), NUMERICAL RECIPES in C++: The Art of Scientific Computing. Second Edition. Editorial Cambridge University Press, 2002.

Jordana, J.R (2002), Ajuste de observaciones. El método de los mínimos cuadrados con aplicaciones a la topografía. Edicions UPC, 2002

Páginas web:

Comandos de C++:
http://www.publispain.com/supertutoriales/programacion/c_y_cplus/cursos/3/index.htm

AGRADECIMIENTOS

Este proyecto no habría podido realizarse sin la ayuda y el apoyo de mis dos tutores: el profesor Albert Prades y el Dr. Felipe Buill. A ellos les debo agradecer que siempre tuvieran tiempo y paciencia para resolver el montón de dudas que se han presentado en todo el proceso.

Dar a Diana Alonso mi mayor agradecimiento por su apoyo y por aguantarme siempre que he estado ausente pensando en este proyecto y escuchando mis teorías pese a no entender ni una palabra de ellas, por ayudarme a editar la memoria, por estar tantos momentos atenta a mi trabajo y por muchas cosas más que no pueden plasmarse en un papel.

A mis amigos Eduard Redaño y a Manuel Fernandez por darme tantos buenos ratos en el laboratorio y por último a cualquiera que se haya interesado en escuchar en qué consistía este proyecto.