Universidad Pública de Navarra

Proyecto final de carrera

Migración de plataforma y evaluación experimental de algoritmos AAM

Autor:
Iñaki Urtasun

Tutor:
Dr. Rafael Cabeza

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Índice general

Índice de figuras **6**

Índice de tablas **7**

1. Objetivo **9**

2. Materiales y métodos **11**
2.1. Método AAM . . . 11
2.1.1. Modelo de forma . . . 11
2.1.2. Modelo de apariencia . . . 12
2.1.3. Instancia de modelo . . . 13
2.1.4. Adecuando AAM . . . 14
2.2. Descripción base de datos GI4E . . . 14
2.3. Método de estudio de la eficacia del software . . . 17
2.3.1. Fase de entrenamiento . . . 17
2.3.2. Fase de optimización . . . 17
2.3.3. Fase de testeo . . . 18

3. Compilación del software original **19**
3.1. Componentes software original . . . 19
3.1.1. Carpeta libaam_code . . . 19
3.1.2. Carpeta compositional-descent . . . 20
3.2. Compilación e instalación en GNU/Linux . . . 21
3.2.1. Compilación código en lenguaje C++ . . . 21
3.2.2. Compilación ficheros MEX . . . 22
3.3. Compilación en Windows . . . 23
3.3.1. Compilación código en lenguaje MATLAB® . . . 24
3.3.2. Compilación código en lenguaje C++ . . . 24
3.4. Verificación del software . . . 33
3.4.1. Pruebas sobre el código en lenguaje MATLAB® . . . 34
3.4.2. Pruebas sobre el código en lenguaje C++ . . . 36

4. Adaptación software MATLAB® a la base de datos GI4E **43**
4.1. Gestión de la base de datos GI4E . . . 43
4.1.1. Programas . . . 44
4.1.2. Funciones . . . 45
4.1.3. Estructuras de datos . . . 47
4.2. Fase de entrenamiento . . . 48
4.2.1. Programas . . . 48
4.2.2. Funciones . . . 49
4.2.3. Estructuras de datos . . . 50

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

4.3. Fase de optimización . . . 54
4.3.1. Programas . . . 54
4.3.2. Funciones . . . 56
4.3.3. Estructuras de datos . . . 57
4.4. Fase de testeo . . . 58
4.4.1. Programas . . . 59
4.4.2. Funciones . . . 61
4.4.3. Estructuras de datos . . . 62

5. Análisis de los resultados 65
5.1. Fase de optimización . . . 65
5.2. Fase de testeo . . . 75
5.2.1. Elección mejores AAM por método . . . 76
5.2.2. Consumo de tiempo y memoria RAM . . . 78

6. Conclusiones 81

7. Bibliografía 83

A. Software utilizado 85
A.1. GNU/Linux - Ubuntu 14.04 LTS 64 bits . . . 85
A.1.1. Instalación compiladores y librerías necesarias . . . 85
A.1.2. Instalación MATLAB® R2011a y R2014a . . . 86
A.2. Windows 7 Professional 64 bits . . . 86
A.2.1. Instalación Microsoft Visual Studio 2013 Professional . . . 86
A.2.2. Instalación MATLAB® R2014a . . . 87
A.2.3. Obtención librerías necesarias . . . 87

B. Especificaciones hardware equipo utilizado 91

C. Funciones de MATLAB® del software original 93

D. Fichero log AAM generado en fase entrenamiento 97

E. Uso interfaz gráfica (GUI) 109
E.1. AAM . . . 109
E.2. BBDD . . . 110
E.3. Fase de entrenamiento . . . 111
E.4. Fase de testeo . . . 113

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Índice de figuras

2.1. Ejemplo modelo de forma . . . 12
2.2. Ejemplo modelo de apariencia . . . 13
2.3. Ejemplo instancia de modelo . . . 13
2.4. Distribución y numeración de los puntos de control marcados manualmente . . . 16
2.5. Fases método de estudio de la eficacia del software . . . 17

3.1. Contenido descomprimido del fichero compos.zip . . . 19
3.2. Contenido de la carpeta libaam_code . . . 19
3.3. Contenido de la carpeta compositional-descent . . . 20
3.4. Contenido de la carpeta proyecto terminada la compilación . . . 23
3.5. Especificaciones generales proyecto MSVS libaam_code . . . 26
3.6. Especificaciones C/C++ proyecto MSVS libaam_code . . . 26
3.7. Especificaciones bibliotecario proyecto MSVS libaam_code . . . 27
3.8. Especificaciones generales proyecto MSVS aam-fitter-cli . . . 29
3.9. Especificaciones C/C++ proyecto MSVS aam-fitter-cli . . . 29
3.10. Especificaciones vinculador proyecto MSVS aam-fitter-cli . . . 30
3.11. Especificaciones generales proyecto MSVS aam-inspector . . . 31
3.12. Especificaciones C/C++ proyecto MSVS aam-inspector . . . 31
3.13. Especificaciones vinculador proyecto MSVS aam-inspector . . . 31
3.14. Resultado obtenido en windows del programa experiment_fit_imm.m sobre el usuario 1 . . . 35
3.15. Contenido de la carpeta de pruebas para el SO GNU/linux . . . 36
3.16. Posicionamiento de AAM sin ajustar . . . 38
3.17. Ajuste de AAM siguiendo el método CoDe . . . 38
3.18. Contenido de la carpeta de pruebas para el SO Windows 7 Professional . . . 40

4.1. Organigramas programas que gestionan la BBDD . . . 45
4.2. Organización interna userdata . . . 47
4.3. Estructura interna AAM . . . 51
4.4. Organigrama p_2_training . . . 53
4.5. Organigramas programas fase optimización . . . 55
4.6. Organización interna ImagesDATA fase optimización . . . 58
4.7. Organigrama p_5_testing.m . . . 60
4.8. Organigramas programas representación fase testeo . . . 61
4.9. Organización interna ImagesDATA fase testeo . . . 63

5.1. Boxplot distancia euclídea (píxeles).Fase optimización, por parámetros . . . 65
5.2. Boxplot distancia euclídea (píxeles).Fase optimización, método LinCoLiNe . . . 67
5.3. Boxplot distancia euclídea (píxeles).Fase optimización, método CoLiNe . . . 68
5.4. Boxplot distancia euclídea (píxeles).Fase optimización, método LinCoDe . . . 70
5.5. Boxplot distancia euclídea (píxeles).Fase optimización, método CoDe . . . 72

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

5.6. Boxplot distancia euclídea (píxeles).Fase optimización, método CoNe 74
5.7. Boxplot AAM óptimos por método 78

E.1. GUI AAM 110
E.2. GUI BBDD 111
E.3. GUI training 112
E.4. GUI testing, pantalla inicial 113
E.5. GUI testing, carga del fichero AAM 114
E.6. GUI testing, selección de una imagen de usuario 115
E.7. GUI testing, ajuste por el método LinCoDe sobre la imagen 2 del usuario 027 . . 116

Índice de tablas

2.1. Diferencias algoritmos AAM implementados 14

4.1. Lista de AAM creados en la fase de entrenamiento 52

5.1. Distancia euclídea (píxeles). Fase optimización, método LinCoLiNe 66
5.2. Distancia euclídea (píxeles). Fase optimización, método CoLiNe 69
5.3. Distancia euclídea (píxeles). Fase optimización, método LinCoDe 71
5.4. Distancia euclídea (píxeles). Fase optimización, método CoDe 73
5.5. Distancia euclídea (píxeles). Fase optimización, método CoNe 75
5.6. Distancia euclídea (píxeles). Fase testeo, método LinCoLiNe 76
5.7. Distancia euclídea (píxeles). Fase testeo, método CoLiNe 76
5.8. Distancia euclídea (píxeles). Fase testeo, método LinCoDe 76
5.9. Distancia euclídea (píxeles). Fase testeo, método CoDe 76
5.10. Distancia euclídea (píxeles). Fase testeo, método CoNe 77
5.11. Fase testeo, AAM óptimos 77
5.12. Tiempo (s) por método 79
5.13. Memoria utilizada (MB) para cargar AAM por método 79
5.14. Memoria utilizada (MB) por método 79

6.1. Clasificación mejores métodos por parámetros 82

B.1. Especificaciones técnicas equipo de pruebas 91

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Capítulo 1

Objetivo

Consiste en migrar el software AAM (modelo activo de apariencia), desarrollado por los profesores Brian Amberg, Andrew Blake y Thomas Vetter, al Sistema Operativo Windows, adaptarlo para poder utilizarlo sobre la base datos de caras GI4E, desarrollada por el departamento de Ingeniería Eléctrica y Electrónica de la Universidad Pública de Navarra, y verificar cuál de los distintos algoritmos implementados resulta más eficiente para dicha base de datos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Capítulo 2

Materiales y métodos

2.1. Método AAM

Un modelo activo de apariencia (AAM, Active Appearance Model) es un algoritmo de visión por ordenador que contiene modelos separados de la forma y de la apariencia (textura) de un objeto. Sirve para identificar cualquier objeto presente en una imagen y su uso está muy extendido en el reconocimiento facial.
El proceso completo de análisis de una imagen mediante AAM, está constituido por dos fases principales:

1. Fase de construcción del modelo AAM (entrenamiento).
   En esta fase se debe crear los modelos necesarios que se utilizarán en la fase de reconocimiento. Esta etapa se realiza una sola vez, ya que una vez se ha obtenido el modelo, se puede utilizar tantas veces como sea necesario. A su vez se puede dividir en los siguientes pasos:

   a) Elección de las imágenes de entrenamiento, que previamente han sido marcadas con los puntos de control (landmarks) adecuados.

   b) Alineamiento y normalización del conjunto de imágenes de entrenamiento.

   c) Cálculo del modelo estadístico. Habitualmente se utiliza PCA (Análisis de Componentes Principales) para este fin.

2. Fase de ajuste del modelo AAM (reconocimiento).
   Se parte del modelo obtenido en la fase anterior y se intenta ajustarlo sobre el objeto a estudiar presente en la imagen a analizar. Se requiere que el contorno del modelo se halle próximo al contorno del objeto. Los pasos a realizar son los siguientes:

   a) La forma o contorno medio de nuestro modelo será proyectado sobre la imagen.

   b) En sucesivas iteraciones se modifican los parámetros de los modelos, dentro de unos límites establecidos, de manera que nunca deje de ser un ejemplo válido. En este proceso se busca la minimización del error entre la imagen y el ajuste AAM.

   c) Cuando acabe este proceso de ajuste, el resultado será una realización del modelo con unos parámetros únicos de forma y apariencia, que se adaptará lo máximo posible a la imagen origen.

2.1.1. Modelo de forma

El modelo de forma de un AAM independiente está definido por una malla y, en particular, por las ubicaciones de los vértices de la misma. Se puede expresar de dos maneras equivalentes:

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

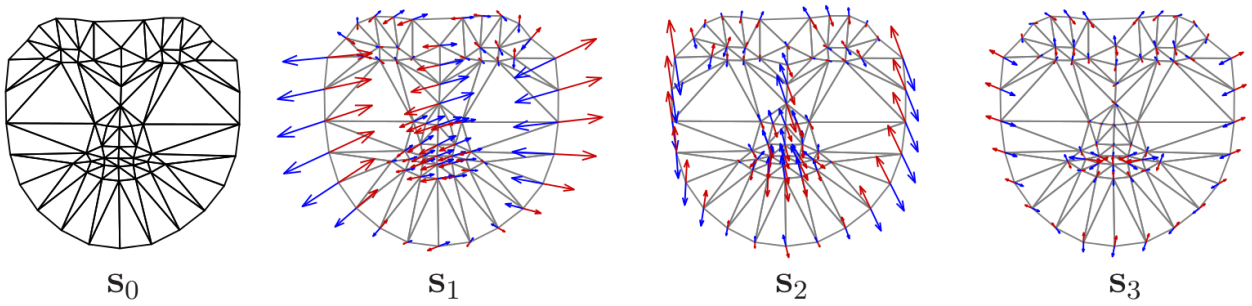


Figura 2.1: Ejemplo modelo de forma

- Como las coordenadas de los vértices que la componen.

$$S = (x_1, y_1, x_2, y_2, ..., x_v, y_v)^T \tag{2.1}$$

- Como una forma básica S_0 más una combinación lineal de n vectores de forma S_i. En este caso los coeficientes p_i son parámetros de forma y suponemos que los vectores S_i son ortonormales.

$$S = S_0 + \sum_{i=1}^{n} p_i S_i \tag{2.2}$$

Se calcula a partir de una serie de imágenes de entrenamiento en la que han sido marcados manualmente los puntos de referencia. Esas mallas marcadas manualmente suelen sufrir un proceso de normalización mediante un análisis de Procrustres (transformación euclidiana que conserva la forma, para eliminar así las diferencias de traslación, rotación y escala entre ellas, ya que se llevan a un marco de referencia común) para posteriormente aplicar sobre ellas la técnica estadística PCA (Análisis de Componentes Principales).
Un ejemplo de modelo de forma se muestra en la figura 2.1. En la parte izquierda de la misma, está representada la forma básica S_0 (68 vértices en el ejemplo). En el resto de la figura, la forma básica S_0 se superpone con las flechas correspondientes a cada uno de los vectores S_1, S_2 y S_3.

2.1.2. Modelo de apariencia

El modelo de apariencia (textura) de un AAM se define por el contenido de la malla S_0 descrita en el apartado anterior. Se puede expresar de dos maneras distintas equivalentes:

- El conjunto de píxeles (imagen) que se encuentran dentro de la malla S_0

$$A(x) = (x, y)^T \tag{2.3}$$

- Como la imagen de apariencia básica $A_0(x)$ más una combinación lineal de m imágenes de apariencia $A_i(x)$. En este caso los coeficientes λ_i son parámetros de apariencia y suponemos que las imágenes $A_i(x)$ son ortonormales.

$$A(x) = A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x) \qquad \forall x \in S_0 \tag{2.4}$$

El modelo se calcula con una serie de imágenes de entrenamiento que son normalizadas a partir de la deformación de la malla S_0. Posteriormente sobre estas imágenes se aplica PCA.
Un ejemplo de modelo de apariencia se muestra en la figura 2.2. En la parte izquierda de la misma está representada la forma básica $A_0(x)$. En el resto de la figura se pueden observar las 3 imágenes de apariencia $A_1(x)$, $A_2(x)$ y $A_3(x)$.

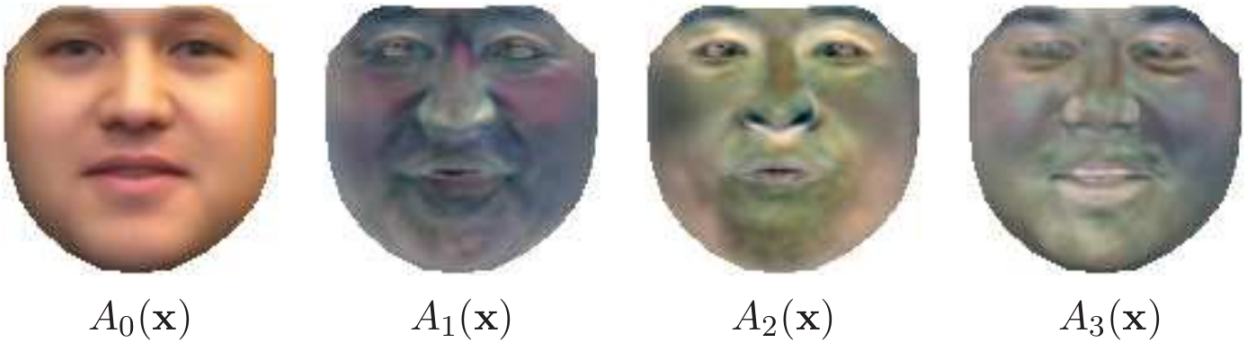


Figura 2.2: Ejemplo modelo de apariencia

2.1.3. Instancia de modelo

Partiendo de los de los parámetros de forma $p = (p_1, p_2, ..., p_n)^T$, utilizando la ecuación 2.2, podemos generar la forma S del AAM. Análogamente, utilizando la ecuación 2.4 y partiendo de los parámetros de apariencia $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m)^T$, podemos generar la imagen $A(x)$ definida en el interior de la malla S_0.
Las mallas S y S_0 están relacionadas y se puede definir una distorsión por partes desde S_0 a S. Cualquier triángulo presente en S_0 posee su correspondiente en S. Para obtener la deformación completa, para cada píxel x presente en S_0 tenemos que encontrar a que triángulo pertenece y entonces deformarlo con la distorsión afín correspondiente de S. A esto lo denotamos como $W(x; p)$.
La instancia de modelo $M(W(x; p)) = A(x)$, con los parámetros de forma p y los parámetros de de apariencia λ, se crea distorsionando la imagen de apariencia A desde la malla S_0 hasta la forma S. Se puede ver un ejemplo en la figura 2.3.

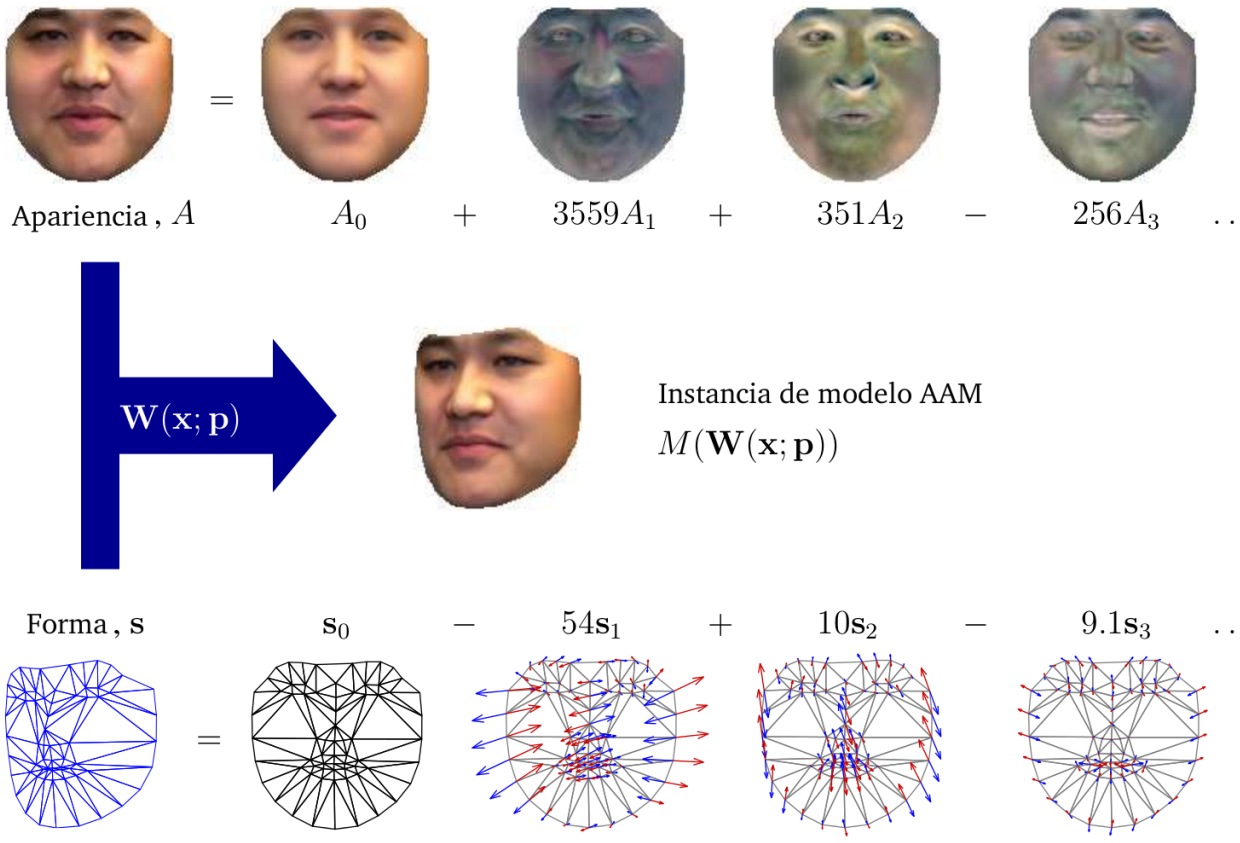


Figura 2.3: Ejemplo instancia de modelo

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

2.1.4. Adecuando AAM

Supongamos que se nos proporciona una imagen de entrada $I(x)$ que queremos adecuar mediante AAM y conocemos los parámetros óptimos tanto de forma (p) como de apariencia (λ). En este caso la imagen $I(x)$ y la instancia de modelo $M(W(x;p)) = A(x)$ tienen que ser similares, y en caso de que no lo sean, la diferencia (error) entre ellas debe ser la mínima posible.
Si x es un píxel presente en S_0, el píxel correspondiente en la imagen de entrada I es $W(x;p)$. En el píxel x, AAM tiene la apariencia $A(x) = A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x)$. En el píxel $W(x;p)$ la imagen de entrada tiene la intensidad $I(W(x;p))$. La siguiente ecuación representa la suma de cuadrados de la diferencia entre ambas:

$$\sum_{x \in S_0} \left[A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x) - I(W(x;p)) \right]^2 \tag{2.5}$$

El objetivo que se busca en los ajustes AAM es minimizar el resultado de la ecuación anterior respecto a los parámetros de forma p y los parámetros de apariencia λ simultáneamente. En general la optimización no es lineal para los parámetros de forma aunque si lo es para los parámetros de apariencia.
Para conseguir este objetivo se utilizan distintas técnicas matemáticas, tales como:

- Matriz Jacobina
- Matriz Hessiana
- Gradiente descendente
- Gradiente ascendente
- ...

El software que debemos adaptar implementa 5 algoritmos diferentes que intentan, mediante la combinación de esas técnicas, ser lo más eficientes, tanto en velocidad como en acierto, posibles. En la tabla 2.1 podemos ver en que se diferencian las distintas implementaciones.

	LinCoLiNe	**CoLiNe**	**LinCoDe**	**CoDe**	**CoNe**
Matriz Hessiana	Gauss-Newton	Aproximada	No usada	No usada	Aproximada
Gradiente descendente	Aproximado	Exacto	Aproximado	Exacto	Exacto

Tabla 2.1: Diferencias algoritmos AAM implementados

Cabe destacar que los métodos LinCoDe y CoDe son nuevos métodos que se introdujeron con la publicación del artículo "On Compositional Image Alignment with an Application to Active Appearance Models" que acompaña al software que tenemos que migrar.

2.2. Descripción base de datos GI4E

La base de datos GI4E contiene imágenes de 115 usuarios, siendo válidas las de los 103 primeros. Las imágenes están tanto en baja resolución (800x600 píxeles) como en alta resolución (2048x1536 píxeles). Nos vamos a centrar en el contenido en alta resolución, ya que estas son las imágenes que vamos a utilizar.
El contenido principal de la base de datos es el siguiente:

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- BBDD_800x600
  - marcas: contiene las marcas, en ficheros de texto, de las imágenes en baja resolución.
  - 001_01.png: imagen 01 del usuario 001.
  - 001_02.png: imagen 02 del usuario 001.
  - 001_12.png: imagen 12 del usuario 001.
  - 002_01.png: imagen 01 del usuario 002.
  - .
  - .
  - 115_12.png: imagen 12 del usuario 115.
- BBDD_2048x1536
  - 001: carpeta del usuario 001
    - images
      - 001_01.png: imagen 01 del usuario.
      - 001_02.png: imagen 02 del usuario.
      - 001_03.png: imagen 03 del usuario.
      - 001_04.png: imagen 04 del usuario.
      - 001_05.png: imagen 05 del usuario.
      - 001_06.png: imagen 06 del usuario.
      - 001_07.png: imagen 07 del usuario.
      - 001_08.png: imagen 08 del usuario.
      - 001_09.png: imagen 09 del usuario.
      - 001_10.png: imagen 10 del usuario.
      - 001_11.png: imagen 11 del usuario.
      - 001_12.png: imagen 12 del usuario.
      - 001_13.png: imagen 13 del usuario. Ojos cerrados
    - videos
      - 001_01.png: imagen 01 del usuario.
      - 001_02.png: imagen 02 del usuario.
      - .
      - .
      - .
      - 001_170.png: imagen 170 del usuario.
  - 002: carpeta del usuario 002.
  - .
  - .
  - 102: carpeta del usuario 102.

- 103: carpeta del usuario 103, último usuario válido
- 104: carpeta del usuario 104, primer usuario descartado.
- .
- .
- 115: carpeta del usuario 115, usuario descartado.
- landmarks: contiene los puntos de referencia de los 115 usuarios, 52 puntos marcados por cada imagen. En la figura 2.4 se puede observar su distribución y numeración.
  - Formato de los archivos - versión con iris.doc: explicación de como se han marcado las caras.
  - Landmarks_115_users_9_strokes.mat
  - Usuario_001.txt: contiene los landmarks de las 13 imágenes del usuario 001.
  - Usuario_002.txt: contiene los landmarks de las 13 imágenes del usuario 002.
  - .
  - .
  - Usuario_115.txt: contiene los landmarks de las 13 imágenes del usuario 115.
- Marcas full size

code: código en lenguaje MATLAB® utilizado para el marcado de caras.

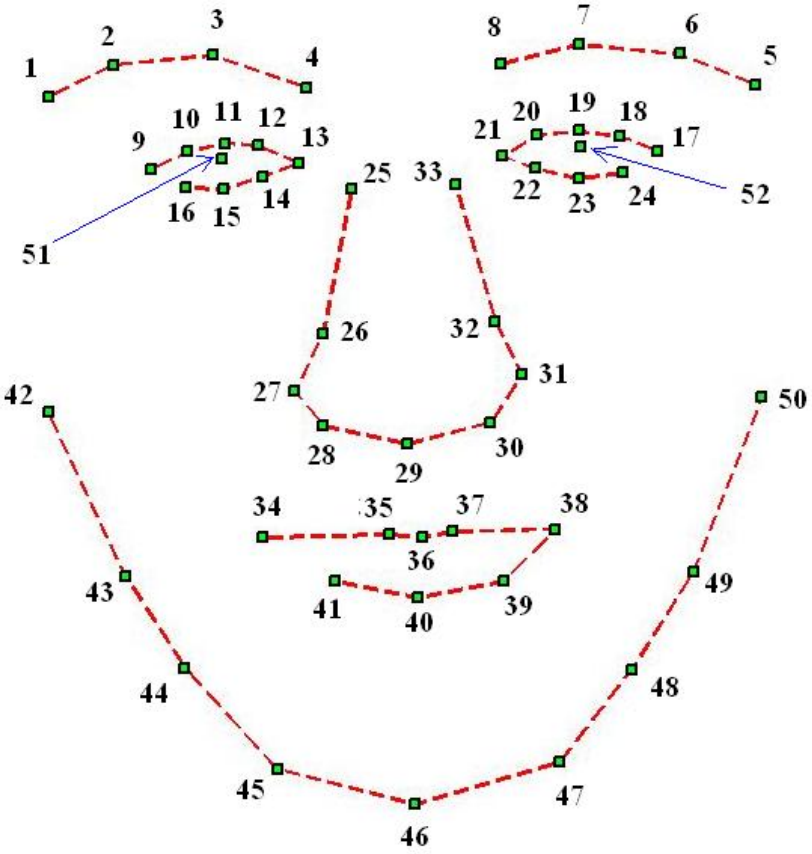


Figura 2.4: Distribución y numeración de los puntos de control marcados manualmente

2.3. Método de estudio de la eficacia del software

Para comprobar la eficacia del software realizaremos pruebas sobre los usuarios de la base de datos GI4E. Escogeremos las imágenes en alta resolución de los mismos, descartando las de baja resolución. Las distintas fases planeadas se pueden observar en la figura 2.5

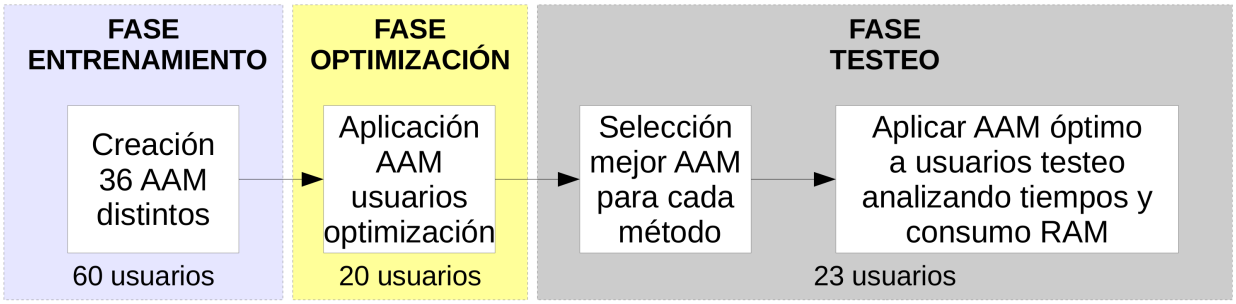


Figura 2.5: Fases método de estudio de la eficacia del software

2.3.1. Fase de entrenamiento

Utilizaremos las imágenes de 60 usuarios de la base de datos para crear una serie de 36 AAM distintos. Los usuarios serán elegidos, de manera aleatoria, de entre los 103 disponibles. De cada uno de los escogidos cogeremos, de manera aleatoria, 8 imágenes de entre las 12 que podemos utilizar (descartamos la imagen número 13 de todos los usuarios). Seleccionaremos sólo 8 imágenes por usuario para intentar aumentar al máximo la variabilidad sin acrecentar demasiado el número de imágenes utilizadas, ya que su incremento implica un mayor uso de memoria.
Las mismas 480 imágenes (8 imágenes de cada uno de los 60 usuarios escogidos) nos servirán para crear todos los AAM de esta fase, variando los parámetros que el software nos permite ajustar para su creación:

- Resolución del modelo: probaremos con resoluciones de 128x128, 192x192 y 256x256 píxeles.
- Tamaño del modelo: probaremos con tamaños de 60, 80 y 100.
- Blur: probaremos activando y desactivando el desenfoque.
- Simetría: nos permite doblar el número de imágenes utilizadas aplicando simetría especular sobre las mismas.

2.3.2. Fase de optimización

En esta fase escogeremos 20 usuarios, elegidos de manera aleatoria entre los 43 restantes de la base de datos. A las 12 imágenes de cada uno de ellos le aplicaremos los 36 AAM obtenidos en la fase anterior, aplicando los distintos métodos de adecuación implementados:

1. Método LinCoLiNe
2. Método CoLiNe
3. Método LinCoDe
4. Método CoDe

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

5. Método CoNe

Estudiaremos la distancia euclídea entre los puntos de referencia marcados manualmente y los puntos calculados tras la aplicación del AAM. Nos quedaremos con los 4 mejores AAM, aquellos que presenten unas menores distancias, para cada uno de los 5 métodos.

2.3.3. Fase de testeo

Seleccionaremos los 23 usuarios restantes de la base de datos, que no han participado en ninguna de las 2 fases anteriores.
A cada una de las 12 imágenes de estos 23 usuarios le aplicaremos los 4 mejores AAM de cada método, escogidos en la fase anterior, hasta obtener el AAM óptimo para un método en concreto para estos usuarios concretos.
Una vez obtenido el AAM óptimo para cada método, procederemos a medir cuanto tiempo y cuanta memoria RAM se utiliza en el proceso. Estas mediciones las realizaremos tanto en el Sistema Operativo Windows como en GNU/Linux, utilizando la versión R2014a del programa MATLAB®. Repetiremos el proceso 2 veces para evitar que pueda haber problemas puntuales (actualizaciones de programas, procesos de mantenimiento...) que falseen las mediciones.

Capítulo 3

Compilación del software original

3.1. Componentes software original

Como podemos leer en el fichero README que acompaña al software, éste consta de dos partes principales: una implementación completa de AAM (modelo activo de apariencia), en lenguaje MATLAB®, y la implementación en lenguaje C++ de los algoritmos de adecuación del método CoDe. La creación de nuevos AAM sólo se pueden realizar con el programa MATLAB®. También se incluye, como ejemplo, un AAM generado con este código.
El software se distribuye en dos carpetas principales, tal y como se puede ver en la figura 3.1:

- libaam_code: contiene el código escrito en lenguaje C++ del método CoDe, así como varios programas para demostrar su utilización.
- compositional-descent: contiene el código fuente de una implementación completa de AAM. Consta de partes escritas en lenguaje MATLAB® así como de código escrito en C++ que se ejecuta dentro del programa MATLAB®. A esta última clase de archivos los llamamos ficheros MEX.

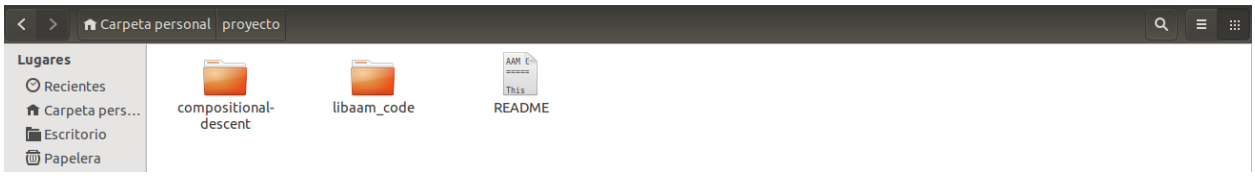


Figura 3.1: Contenido descomprimido del fichero compos.zip

3.1.1. Carpeta libaam_code

El contenido de dicha carpeta es la que se puede observar en la figura 3.2.

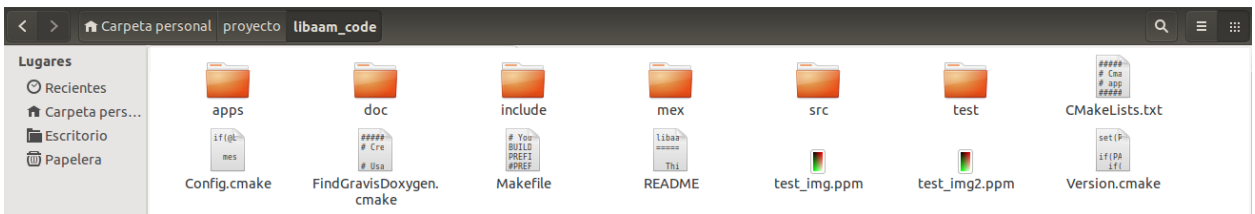


Figura 3.2: Contenido de la carpeta libaam_code

Pasamos a nombrar sus componentes principales:

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- apps/aam-fitter-cli: código fuente de la aplicación aam-fitter-cli.
- apps/aam-inspector: código fuente de la aplicación aam-inspector.
- doc: código para obtener la documentación.
- include: headers necesarios para la compilación del código.
- mex: funciones de MATLAB® escritas en C++, así como algunos programas de ejemplo.
- src: código fuente de la librería.
- test: código fuente de algunos programas de test.

3.1.2. Carpeta compositional-descent

La carpeta que contiene el código MATLAB® tiene el contenido que se puede ver en la figura 3.3.

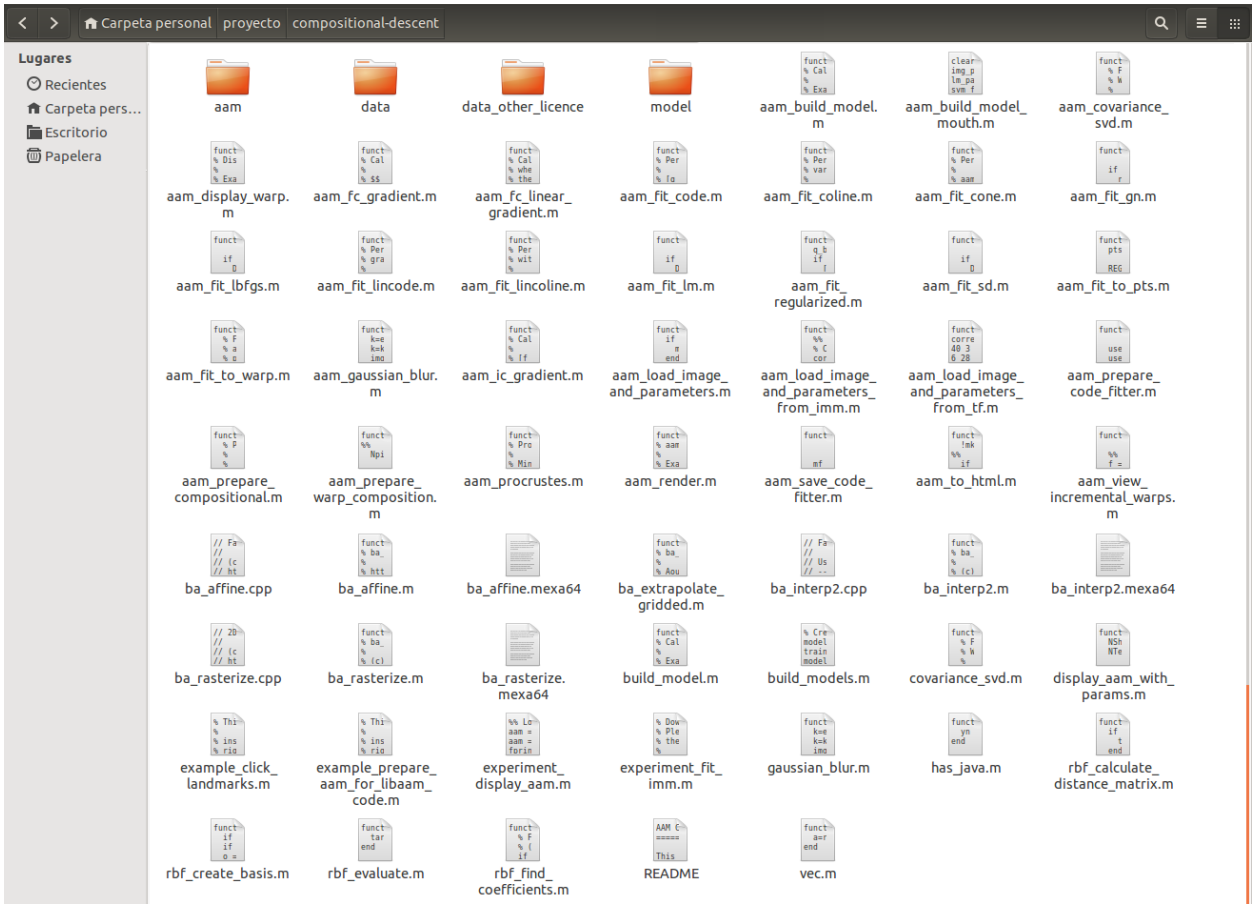


Figura 3.3: Contenido de la carpeta compositional-descent

Pasamos a nombrar las carpetas y programas principales:

- aam: carpeta que contiene el AAM utilizado en los programas de ejemplo.
- data/mixed_aam: carpeta que contiene los ficheros de landmarks (ficheros .landmarks), las imágenes de usuario (ficheros.png) y las máscaras de recorte (*_mask.png) necesarios para los programas de ejemplo. Las imágenes de usuario están corruptas (ocupan 0 bytes).
- data_other_license: carpeta vacía.

- model: carpeta vacía.
- aam_build_model_mouth.m: programa para la creación de un AAM especializado en el contorno de la boca (no lo he podido probar al faltar las imágenes necesarias).
- build_models.m: programa para la creación de un AAM.
- example_click_landmarks.m: programa para crear los landmarks necesarios para poder utilizar con el programa aam-fitter-cli (programa en lenguaje C++).
- example_prepare_aam_for_libaam_code.m: programa para pasar el AAM del formato de programa MATLAB® al formato necesario por la librería en C++.
- experiment_display_amm.m: programa para mostrar el contenido de un AAM.
- experiment_fit_imm.m: programa para comprobar el funcionamiento de la implementación sobre la base de datos de caras IMM de la Universidad Técnica de Dinamarca.

En el Anexo C "Funciones de MATLAB® del software original" hay un listado completo de todos los ficheros presentes en esta carpeta, así como su función.

3.2. Compilación e instalación en GNU/Linux

Atendiendo a la documentación que acompaña el software, para compilar el código fuente es necesario disponer de un Sistema Operativo GNU/Linux 64 bits que disponga del software siguiente:

- Librerías BLAS
- Librerías LAPACK
- Librerías BOOST (filesystem, program_options, system)
- Librerías fftw3
- Cmake
- Programa MATLAB®

Nos hemos decantado por utilizar la distribución Ubuntu 14.04 LTS de 64 bits con el programa MATLAB® R2011a, aunque luego hemos tenido que utilizar la versión R2014a. En el anexo A.1 "Software utilizado-GNU/Linux Ubuntu 14.04 LTS 64 bits" se puede ver el método para instalar el software necesario.

3.2.1. Compilación código en lenguaje C++

El proceso de compilación es el siguiente:

1. Accede a la carpeta donde esté situada la carpeta libaam_code:

   ```
   $ cd libaam_code
   ```

2. Edita el fichero Makefile:

```
$ nano Makefile
```

3. Modifica los apartados BUILD_TYPE, PREFIX y BUILD_DIR. Por ejemplo:

```
BUILD_TYPE=Release
PREFIX=/home/iurtasun/proyecto/install
BUILD_DIR=/home/iurtasun/proyecto/build
```

4. Salva los cambios pulsando las teclas CTRL+X.

5. Una vez acondicionado el fichero makefile ejecuta en la consola:

```
$ make
```

La compilación tarda unos minutos. Una vez terminada obtendremos en la carpeta establecida en el parámetro PREFIX (install en nuestro caso) la siguiente estructura de carpetas:

- bin
  - aam-fitter-cli: aplicación.
  - aam-inspector: aplicación.
- include: contiene los headers utilizados.
- lib
  - libaam_code-1.so: enlace hacia la librería obtenida.
  - libaam_code-1.so.1: librería obtenida.
- matlab/aam_code_mex-1: contienen los ficheros MEX compilados para su uso en MATLAB®.
- share
  - aam_code-1: contiene la configuración utilizada por cmake para la compilación.
  - doc: contiene toda la documentación sobre la librería

Para su instalación sólo hay que copiar la estructura de carpetas obtenida a la carpeta /usr, añadiendo el contenido al ya existente: copiar aam-fitter-cli a /usr/bin, aam-inspector a /usr/bin, libaam_code-1.so a /usr/lib...

3.2.2. Compilación ficheros MEX

En el apartado anterior se compilan la mayoría de los ficheros MEX, pero aún faltan por compilar 3 de ellos localizados en la carpeta compositional-descent (como se puede observar en la figura 3.3 ya existen versiones compiladas de los ficheros MEX, pero es mejor volverlos a crear para asegurarnos de que utilizan las librerías correctas de nuestro SO).
Como todo el código dentro de la carpeta compositional-descent está en lenguaje MATLAB® renombramos dicha carpeta a matlab. Dentro de la carpeta renombrada creamos una subcarpeta llamada mex y copiamos el contenido de la carpeta install/matlab/aam_code_mex-1 obtenida en la compilación del apartado anterior.
Una vez ya tenemos todo correctamente organizado pasamos a escoger el compilador a utilizar por el programa MATLAB® para la compilación de los ficheros MEX restantes. Para ello:

1. Abre un terminal.
2. Ejecuta el comando:

   ```
   $ mex -setup C++
   ```

   Así escogemos el compilador g++ que utilizaremos para compilar los ficheros MEX.

Una vez tenemos ya escogido el compilador a utilizar por MATLAB® tenemos que ir compilando manualmente cada uno de los archivos, ejecutando en un terminal los comandos necesarios.

1. Abre un terminal.
2. Accede a la carpeta donde están los ficheros a compilar:

   ```
   $ cd matlab
   ```

3. Para compilar el fichero ba_affine.cpp ejecuta:

   ```
   $ mex -v ba_affine.cpp
   ```

   Se obtiene el fichero ba_affine.mexa64
4. Para compilar el fichero ba_interp2.cpp ejecuta:

   ```
   $ mex -v ba_interp2.cpp
   ```

   Se obtiene el fichero ba_interp2.mexa64
5. Para compilar el fichero ba_rasterize.cpp ejecuta:

   ```
   $ mex -v ba_rasterize.cpp
   ```

   Se obtiene el fichero ba_rasterize.mexa64

Con esto ya hemos terminado la compilación de los programas originales en el SO en GNU/Linux. Si hemos seguido todas las instrucciones tendremos la estructura de carpetas que podemos observar en la figura 3.4.

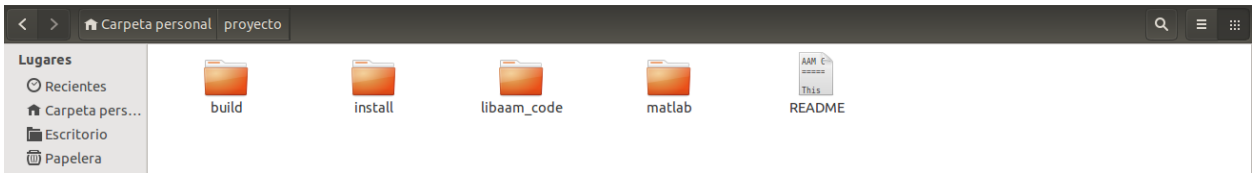


Figura 3.4: Contenido de la carpeta proyecto terminada la compilación

3.3. Compilación en Windows

Una vez comprobado que el software compila y funciona en el Sistema Operativo (SO) en el que fue diseñado tendremos que ajustarlo para que funcione en el SO Windows. Utilizamos la versión Windows 7 Professional de 64 bits.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

3.3.1. Compilación código en lenguaje MATLAB®

El software MATLAB® es multiplataforma, por lo que todo programa escrito en este lenguaje debe funcionar en cualquier SO soportado (salvo que se utilice alguna función exclusiva de un SO concreto, lo que no es el caso). Los ficheros MEX, al ser código escrito en lenguaje C++ ejecutado en MATLAB®, si que hay que compilarlos con el compilador de C++ escogido. Este tema se trata en el apartado 3.3.2.2. Compilación ficheros MEX .

3.3.2. Compilación código en lenguaje C++

El software desarrollado en C y C++ si que presenta mas problemas a la hora de pasar de una plataforma a otra debido a la diversidad de compiladores disponibles, versiones distintas del estándar... Después de probar con distintos compiladores (CyGwin, MinGW) decidimos utilizar el compilador de Microsoft Visual Studio 2013 Professional (MSVS), ya que este software cumple con el estándar C++11, lo que resuelve varios errores de compilación que sucedían con las versiones anteriores. La elección de este compilador implica que debamos usar la versión R2014a, o posterior, del programa MATLAB®.
Como los requisitos siguen siendo los mismos (librerías BLAS, LAPACK...) lo primero que hay que hacer es obtener versiones de dichas librerías para el SO Windows 7 Professional de 64 bits, así como instalar tanto el programa MSVS 2013 como el programa MATLAB®. En el anexo A.2 "Software utilizado-Windows 7 Professional 64 bits"se puede ver los pasos seguidos para su instalación.

3.3.2.1. Compilación librería y programas ejemplo

Creamos una carpeta en la raíz del disco duro C con el nombre MSVC. Dentro creamos 5 subcarpetas:

- aam-fitter-cli: carpeta donde estará el proyecto de MSVS para la aplicación con este nombre.
- aam-inspector: carpeta donde estará el proyecto de MSVS para la aplicación con este nombre.
- include: carpeta donde incluiremos todos los headers necesarios. Tendrá 3 subcarpetas:
  - aam_code: headers del propio código fuente. Copiamos el contenido de la carpeta libaam_code/include del software original.
  - boost: headers de la librerías boost. Los copiamos una vez terminada la compilación de las mismas. Ver apartado A.2.3.
  - fftw: copiamos el fichero fftw3.h que hemos descargado. Ver apartado A.2.3.
- lib: carpeta donde incluiremos todas las librerías necesarias para la compilación obtenidas en el apartado A.2.3. Tendrá 3 subcarpetas:
  - blas:
    - libblas.lib
    - liblapack.lib

- boost: separamos en dos carpetas las distintas versiones (debug y release):
  - debug:
    - libboost_filesystem-vc120-mt-gd-1_57.lib
    - libboost_program_options-vc120-mt-gd-1_57.lib
    - libboost_system-vc120-mt-gd-1_57.lib
  - release:
    - libboost_filesystem-vc120-mt-1_57.lib
    - libboost_program_options-vc120-mt-1_57.lib
    - libboost_system-vc120-mt-1_57.lib
- fftw:
  - libfftw3-3.lib
  - libfftw3-3f.lib
  - libfftw3-3f.lib
- libaam_code: carpeta donde estará el proyecto de MSVS de la librería.

Una vez creada la estructura pasamos a crear los proyectos con el programa MSVS 2013. Realizaremos 3 proyectos distintos: uno para la librería, otro para el programa aam-fitter-cli y otro para aam-inspector

Proyecto libaam_code

En este proyecto crearemos la librería estática que luego utilizarán el resto de programas. Sus especificaciones, tanto en su versión debug como release, se pueden ver en las figuras 3.5, 3.6 y 3.7.

Cuando se crea un proyecto automáticamente se crean un par de carpetas donde hay que colocar el código fuente del mismo.

- src: aquí debe estar el código fuente de la librería, que copiaremos de la carpeta libaam_code/src original, quedando con el siguiente contenido
  - matrix:
    - Assert.cpp
    - ImageOperations.cpp
    - Types.cpp
  - ppmb_io:
    - ppmb_io.cpp
  - aam_code.cpp
- include: para los headers. En este caso no copiaremos nada (ya hemos pasado todo los headers de la librería a la carpeta C:\MSVC\include\aam_code)

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

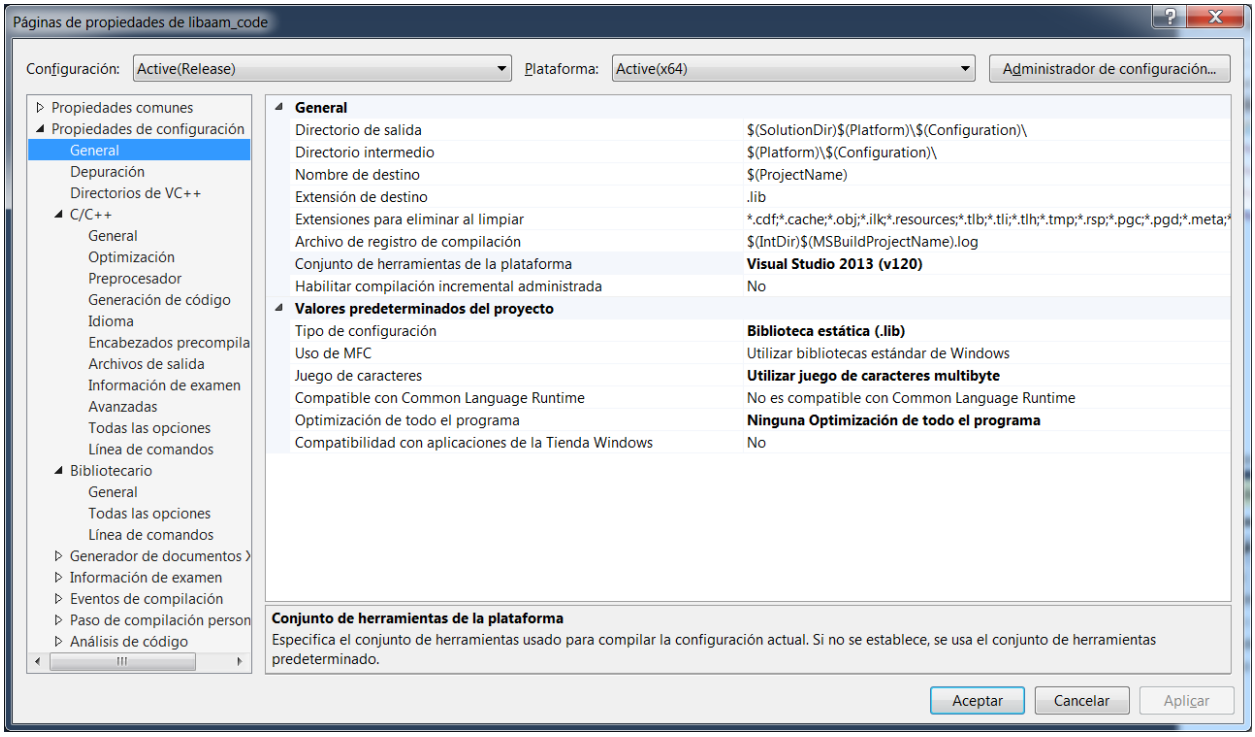


Figura 3.5: Especificaciones generales proyecto MSVS libaam_code

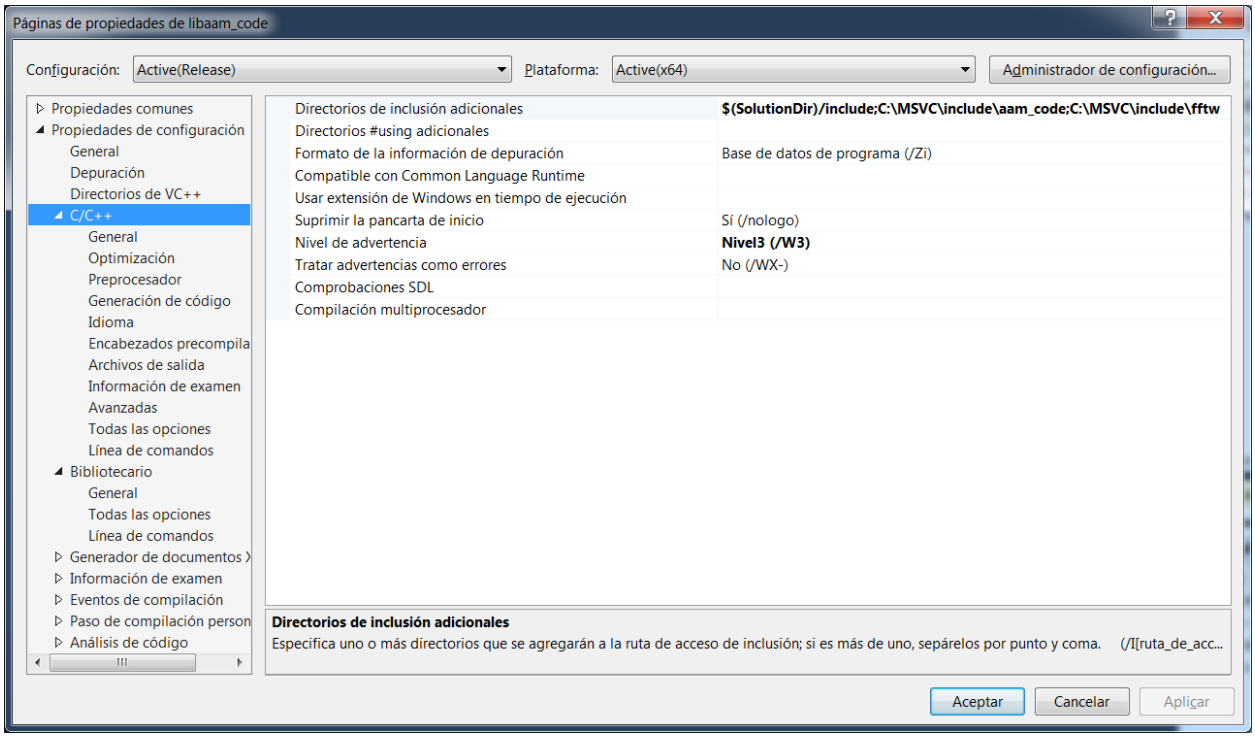


Figura 3.6: Especificaciones C/C++ proyecto MSVS libaam_code

Una vez ya tenemos todo preparado podemos pasar a intentar compilar la librería. Cuando lo hacemos surgen varios errores:

1. Errores con el documento ppmb_io.cpp

```
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(361): error C2679:
 '<<' binario : no se encontró un operador que adopte un operando en la
```

```
 parte derecha de tipo 'std::string' (o bien no existe una conversión
 aceptable)
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(548): error C3861:
 'getline': no se encontró el identificador
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(576): error C2679:
 '<<' binario : no se encontró un operador que adopte un operando en la
 parte derecha de tipo 'std::string' (o bien no existe una conversión
 aceptable)
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(760): error C2679:
 '<<' binario : no se encontró un operador que adopte un operando en la
 parte derecha de tipo 'std::string' (o bien no existe una conversión
 aceptable)
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(1191): error C2784:
 'std::_String_iterator<_Elem,_Traits,_Alloc> std::operator
 +(_String_iterator<_Elem,_Traits,_Alloc>::difference_type,std::_
 String_iterator<_Elem,_Traits,_Alloc>)' : no se pudo deducir el
 argumento de plantilla para 'std::_String_iterator<_Elem,
 _Traits,_Alloc>' desde 'char'
c:\msvc\libaam_code\src\ppmb_io\ppmb_io.cpp(1195): error C2784:
 'std::_String_iterator<_Elem,_Traits,_Alloc> std::operator
 +(_String_iterator<_Elem,_Traits,_Alloc>::difference_type,std::
 _String_iterator<_Elem,_Traits,_Alloc>)' : no se pudo deducir el
 argumento de plantilla para 'std::_String_iterator<_Elem,_Traits
 ,_Alloc>' desde 'char'}
```

Que solucionamos añadiendo en el principio del fichero la cabecera:

```
# include <sstream>
```

Y así incluimos funciones de cadena necesarias, que MSVS no incluye en fstream.

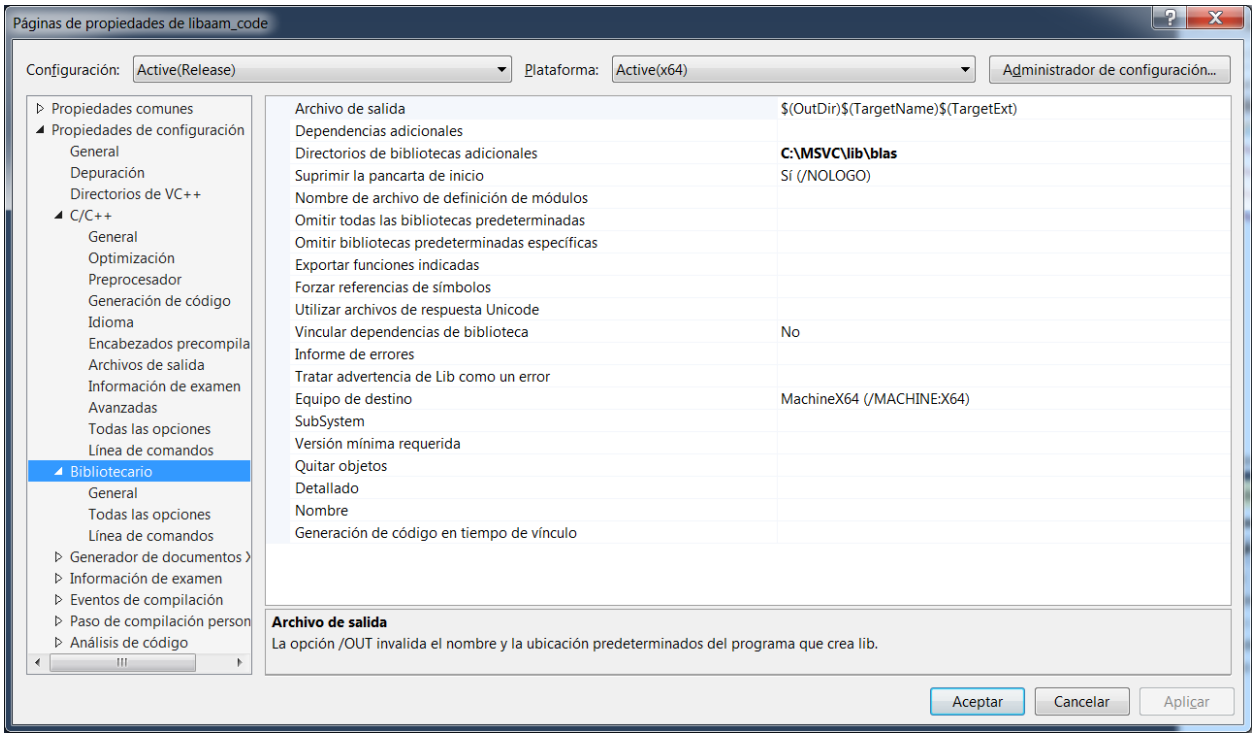


Figura 3.7: Especificaciones bibliotecario proyecto MSVS libaam_code

2. Errores con el documento ImageOperations.cpp

   ```
   c:\msvc\include\libaam_code\matrix\allocator.hpp(69): error C3861:
    'posix_memalign': no se encontró el identificador
   ```

   Para solucionar edita el fichero C:\MSVC\include\libaam_code\matrix\allocator.hpp y sustituye la función AlignedAllocator por el código siguiente (el código original está comentado):

```
Template <class T, size_t alignment = 16>
class AlignedAllocator {
  public:
    static T* alloc(size_t numel){
      //void *r;
      //if (posix_memalign(&r, alignment, (numel == 0 ? 1 : numel)*
          sizeof(T)) != 0)        throw std::runtime_error ("Could not
          allocate memory.");
      //return (T*)r;
      return (T*) ::_aligned_malloc((numel == 0 ? 1 : numel)*sizeof(T
          ),alignment);
    }
    static T* realloc(T *prev, size_t numel) {
      //free(prev);
      _aligned_free(prev);
      return alloc(numel);
    }
    static void free(T *data){
      //::free(data);
      ::_aligned_free(data);
    }
};
```

Una vez resueltos los problemas ya compilará el proyecto sin problemas. Crearemos tanto la versión debug como la versión release de la librería:

- C:\MSVC\libaam_code\x64\debug\libamm_code.lib: versión debug de la librería.
- C:\MSVC\libaam_code\x64\release\libamm_code.lib: versión release de la librería.

La utilizaremos en los proyectos siguientes para la creación de las aplicaciones.

Proyecto aam-fitter-cli

En este proyecto crearemos la aplicación aam-fitter-cli. Sus especificaciones en su versión release, se pueden ver en las figuras 3.8, 3.9 y 3.10. Las especificaciones para la versión debug son prácticamente iguales. Sólo hay que modificar en el apartado vinculador la ubicación de las librerías a utilizar, especificando que sean las versiones debug de las mismas.
Cuando se crea un proyecto automáticamente se crean un par de carpetas donde hay que colocar el código fuente del mismo.

- src: aquí copiamos el código fuente desde la carpeta libaam_code/app/aam-fitter-cli/src original.
  - main.cpp

include: aquí copiamos los headers desde la carpeta libaam_code/apps/aam-fitter-cli/include original.

program_options.hpp

Una vez ya tenemos todo preparado podemos pasar a compilar el programa. Se realiza sin ningún error, obteniendo tanto la versión debug como la versión release de la aplicación:

C:\MSVC\aam-fitter-cli\x64\debug\aam-fitter-cli.exe: versión debug de la aplicación.

C:\MSVC\aam-fitter-cli\x64\release\aam-fitter-cli.exe: versión release de la aplicación.

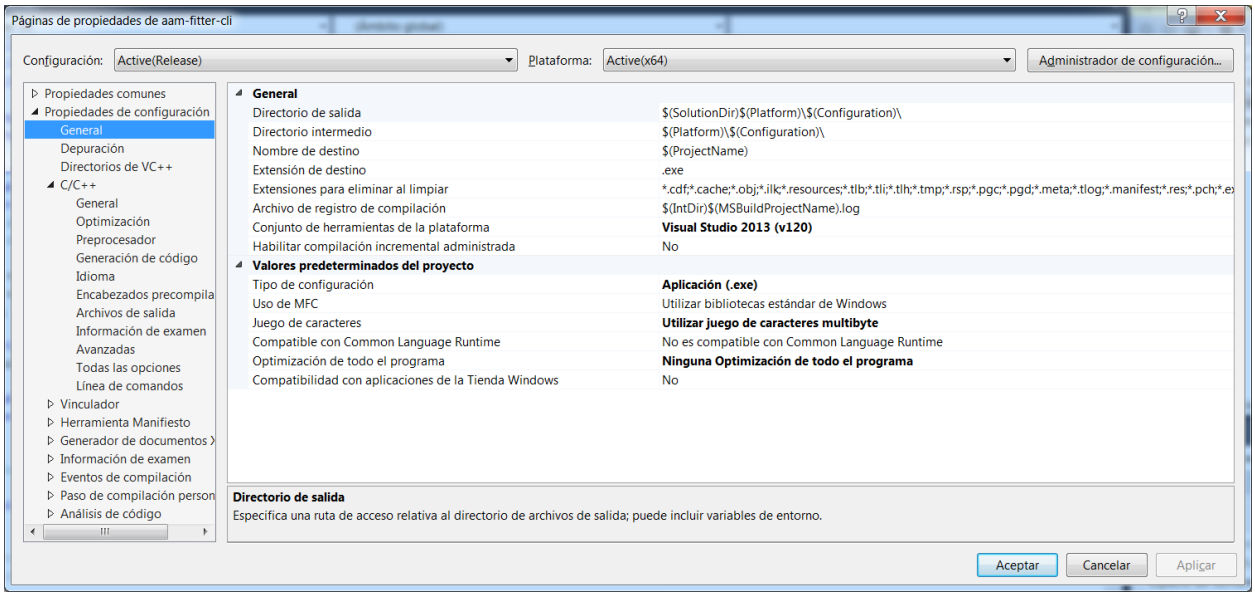


Figura 3.8: Especificaciones generales proyecto MSVS aam-fitter-cli

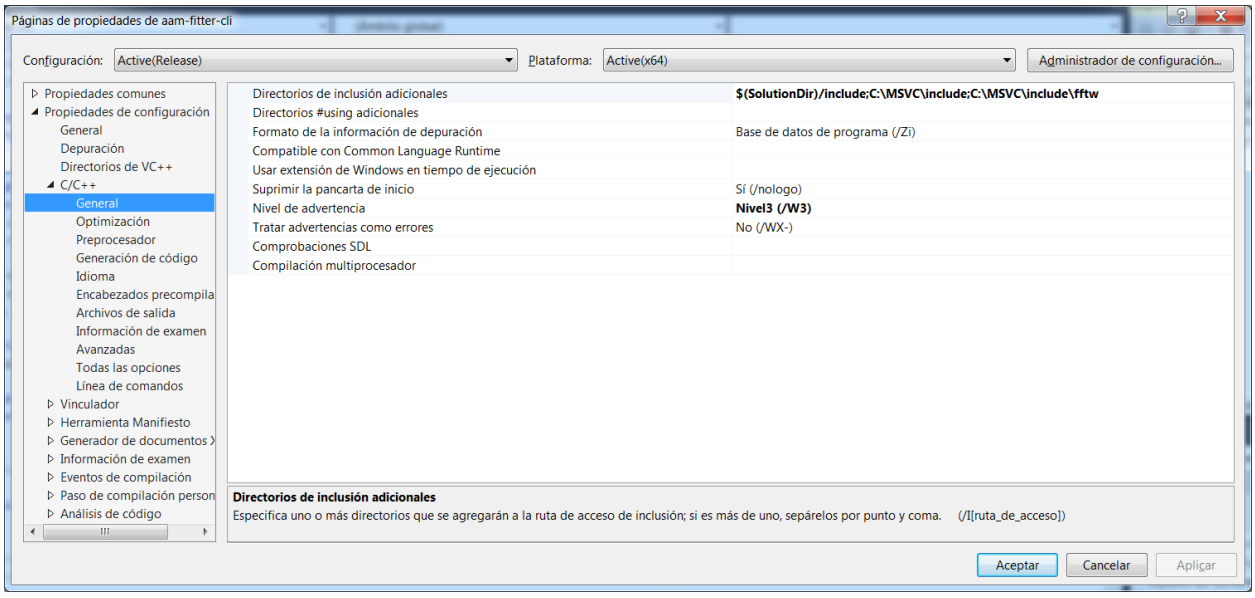


Figura 3.9: Especificaciones C/C++ proyecto MSVS aam-fitter-cli

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

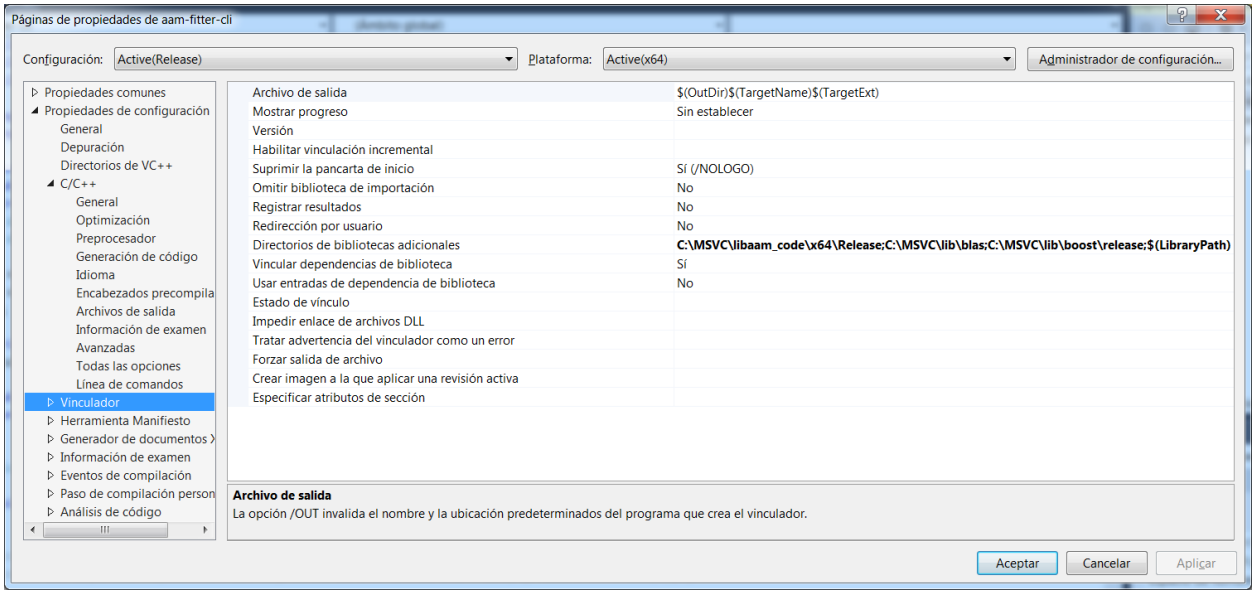


Figura 3.10: Especificaciones vinculador proyecto MSVS aam-fitter-cli

Proyecto aam-inspector

En este proyecto crearemos la aplicación aam-inspector. Sus especificaciones en su versión release, se pueden ver en las figuras 3.11, 3.12 y 3.13. Las especificaciones para la versión debug son prácticamente iguales. Sólo hay que modificar en el apartado vinculador la ubicación de las librerías a utilizar, especificando que sean las versiones debug de las mismas.

Cuando se crea un proyecto automáticamente se crean un par de carpetas donde hay que colocar el código fuente del programa.

- src: aquí copiamos el código fuente desde la carpeta libaam_code/app/aam-inspector/src original.
  - main.cpp
- include: aquí copiamos los headers desde la carpeta program_options.hpp desde la carpeta libaam_code/app/aam-inspector/include original.
  - program_options.hpp

Una vez ya tenemos todo preparado podemos pasar a compilar el programa. Se realiza sin ningún error, obteniendo tanto la versión debug como la versión release de la aplicación:

- C:\MSVC\aam-inspector\x64\debug\aam-inspector.exe: versión debug de la aplicación.
- C:\MSVC\aam-inspector\x64\release\aam-inspector.exe: versión release de la aplicación.

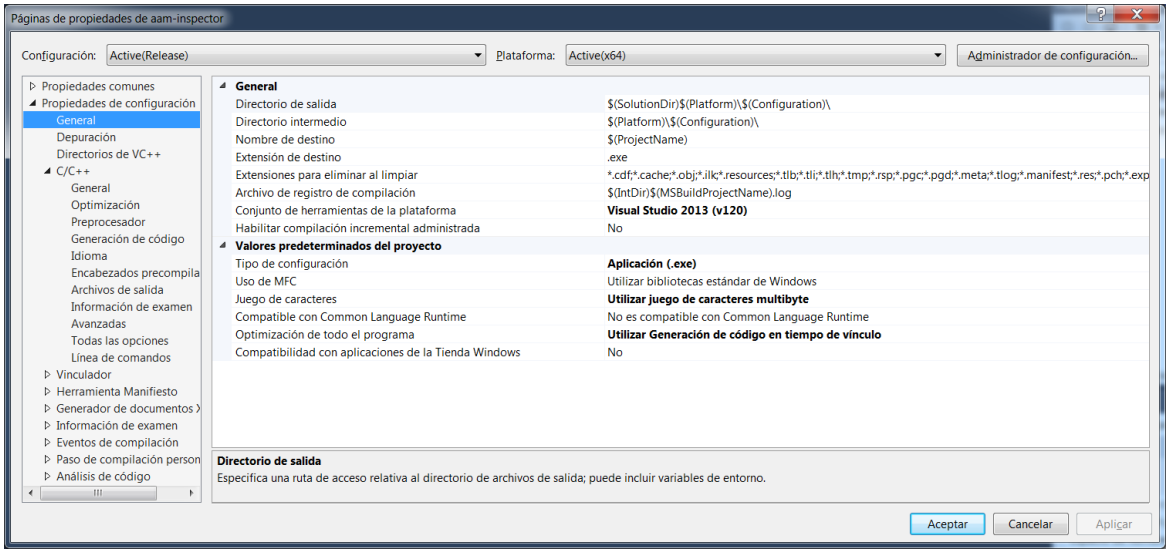


Figura 3.11: Especificaciones generales proyecto MSVS aam-inspector

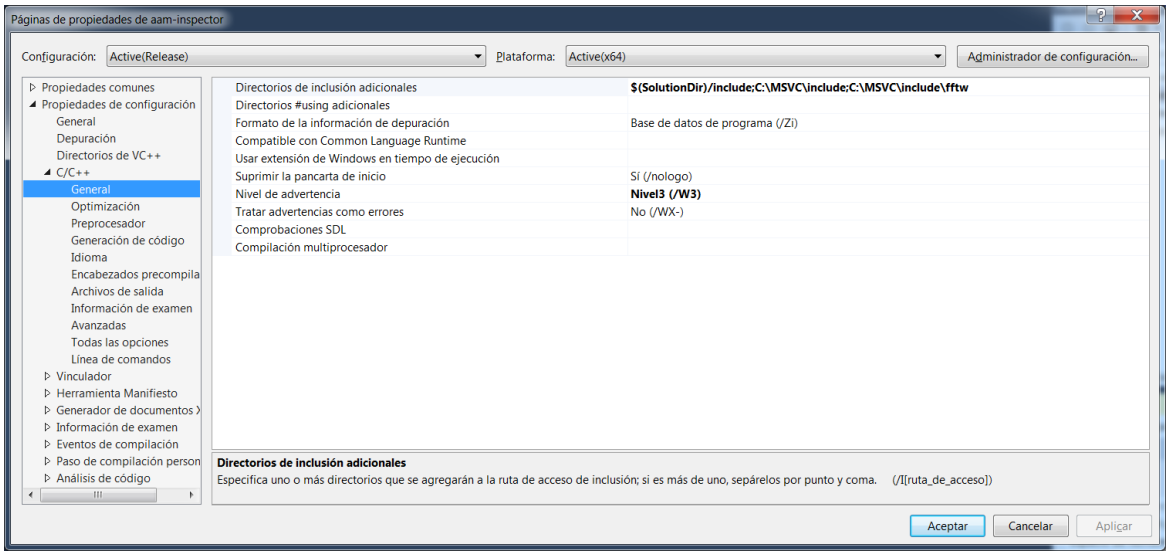


Figura 3.12: Especificaciones C/C++ proyecto MSVS aam-inspector

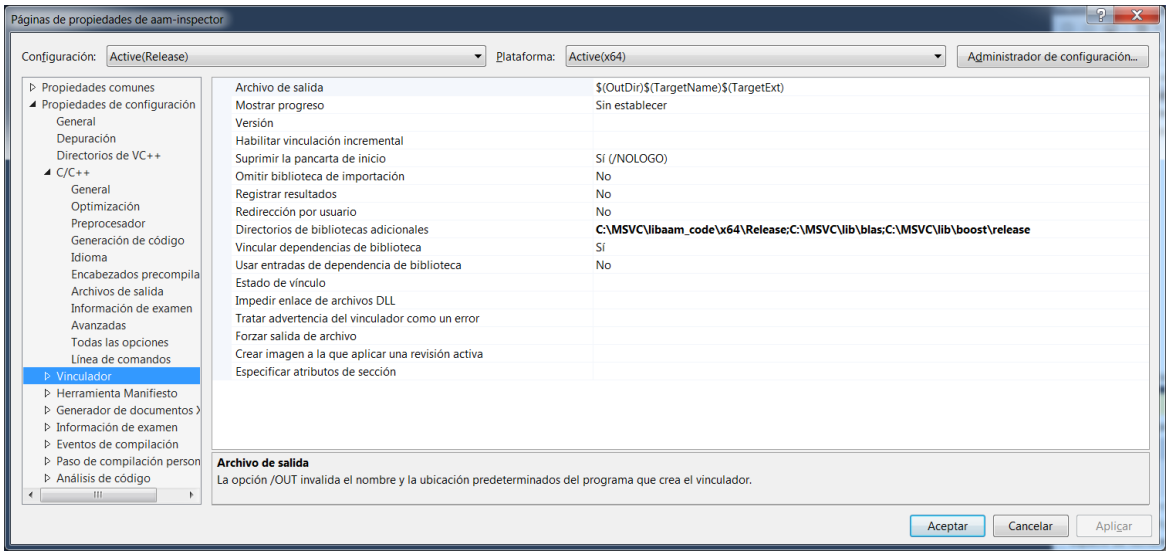


Figura 3.13: Especificaciones vinculador proyecto MSVS aam-inspector

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

3.3.2.2. Compilación ficheros MEX

Lo primero que tenemos que hacer es especificar el compilador que vamos a utilizar. Para ello:

1. Abre un terminal con los compiladores de MS (Símbolo del sistema de las herramientas nativas de VS2013 x64).
2. Ejecuta el comando:

   ```
   > mex -setup
   ```

   Asegurate de escoger el compilador de Microsoft.

Una vez tenemos ya escogido el compilador a utilizar por MATLAB® tenemos que ir compilando manualmente cada uno de los archivos, ejecutando en un terminal los comandos necesarios en el que debemos especificar la ubicación de los headers, librerías... Consideramos la misma estructura de carpetas creada para la compilación de la librería (ver apartado anterior).

1. Abre un terminal.
2. Accede a la carpeta donde están los ficheros a compilar:

   ```
   > cd C:\matlab
   ```

3. Para compilar el fichero ba_affine.cpp ejecuta:

   ```
   > mex -v ba_affine.cpp
   ```

   Se obtiene el fichero ba_affine.mexw64
4. Para compilar el fichero ba_interp2.cpp ejecuta:

   ```
   > mex -v ba_interp2.cpp
   ```

   Se obtiene el fichero ba_interp2.mexw64
5. Para compilar el fichero ba_rasterize.cpp ejecuta:

   ```
   > mex -v ba_rasterize.cpp
   ```

   Se obtiene el fichero ba_rasterize.mexw64
6. Accede a la carpeta donde están el resto de los ficheros a compilar:

   ```
   > cd C:\matlab\mex
   ```

7. Para compilar el fichero aam_fit_code_mex.cpp ejecuta:

   ```
   > mex -IC:\MSVC\include -IC:\MSVC\include\fftw -LC:\MSVC\lib\blas
     -LC:\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
     aam_fit_code_mex.cpp
   ```

   Se obtiene el fichero aam_fit_code_mex.mexw64

8. Para compilar el fichero aam_fit_to_warp_mex.cpp ejecuta:

   ```
   > mex -IC:\MSVC\include -IC:\MSVC\include\fftw -LC:\MSVC\lib\blas
     -LC:\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
     aam_fit_to_warp_mex.cpp
   ```

   Se obtiene el fichero aam_fit_to_warp_mex.mexw64

9. Para compilar el fichero aam_gradient_fc_mex.cpp ejecuta:

   ```
   > mex -IC:\MSVC\include -IC:\MSVC\include\fftw -LC:\MSVC\lib\blas
     -LC:\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
     aam_gradient_fc_mex.cpp
   ```

   Se obtiene el fichero aam_gradient_fc_mex.mexw64

10. Para compilar el fichero aam_load_code.cpp ejecuta:

    ```
    > mex -IC:\MSVC\include -IC:\MSVC\include\fftw  -LC:\MSVC\lib\blas
      -LC\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
      aam_load_code.cpp
    ```

    Se obtiene el fichero aam_load_code.mexw64

11. Para compilar el fichero aam_save_code.cpp ejecuta:

    ```
    > mex -IC:\MSVC\include -IC:\MSVC\include\fftw  -LC:\MSVC\lib\blas
      -LC:\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
      aam_save_code.cpp
    ```

    Se obtiene el fichero aam_save_warp_fitter.mexw64

12. Para compilar el fichero aam_save_warp_fitter.cpp ejecuta:

    ```
    > mex -IC:\MSVC\include -IC:\MSVC\include\fftw -LC:\MSVC\lib\blas
      -LC:\MSVC\libaam_code\x64\Release -llibblas -lliblapack -llibaam_code
      aam_save_warp_fitter.cpp
    ```

    Se obtiene el fichero aam_save_warp_fitter.mexw64

13. Para compilar el fichero ba_interp2.cpp

    ```
    > mex -v ba_interp2.cpp
    ```

    Se obtiene ba_interp2.mexw64

3.4. Verificación del software

Una vez compilado el software en ambas plataformas hay que comprobar que la compilación es correcta, comparando los resultados en ambos Sistemas Operativos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

3.4.1. Pruebas sobre el código en lenguaje MATLAB®

La comprobación en este caso consiste en ejecutar los distintos programas originales y comprobar su resultado.

3.4.1.1. Pruebas en GNU/Linux

Ejecutamos las pruebas en la versión R2011a del programa, haciendo pequeñas modificaciones para ajustar las rutas.

- aam_build_model_mouth.m: No ejecuta ya que faltan las imágenes necesarias para que funcione el programa.
- build_models.m: No ejecuta ya que las imágenes de los usuarios (ficheros .png), disponibles en la carpeta data/mixed_aam, están corruptas (su tamaño es de 0 bytes).
- example_click_landmarks.m: Ejecuta correctamente, creando los ficheros landmarks de la imagen de muestra.
- example_prepare_aam_for_libaam_code.m: Ejecuta correctamente, creando los ficheros:
  - aam_color_mixed_aam_all_192_912_60_1.code
  - aam_color_mixed_aam_all_192_912_60_1.code.lm_fitter

  a partir del AAM de ejemplo aam_color_mixed_aam_all_192_912_60_1.mat.
- experiment_display_amm.m: Ejecuta correctamente hasta el momento que tiene que exportar las imágenes de forma, fallando y dejando la exportación a medias.
- experiment_fit_imm.m: Una vez seguidas las instrucciones (descargada la la BBDD de imágenes que sugiere) ejecuta correctamente.

3.4.1.2. Pruebas en Windows

Ejecutamos las pruebas en la versión R2014a del programa, haciendo pequeñas modificaciones para ajustar las rutas.

- aam_build_model_mouth.m: No ejecuta ya que faltan las imágenes necesarias para que funcione el programa.
- build_models.m: No ejecuta ya que las imágenes de los usuarios (ficheros .png), disponibles en la carpeta data/mixed_aam, están corruptas (su tamaño es de 0 bytes).
- example_click_landmarks.m: Ejecuta correctamente, creando los ficheros landmarks de la imagen de muestra.
- example_prepare_aam_for_libaam_code.m: Ejecuta correctamente, creando los ficheros:
  - aam_color_mixed_aam_all_192_912_60_1.code
  - aam_color_mixed_aam_all_192_912_60_1.code.lm_fitter

  a partir del AAM de ejemplo aam_color_mixed_aam_all_192_912_60_1.mat.

- experiment_display_amm.m: Ejecuta correctamente hasta el momento que tiene que exportar las imágenes de forma, fallando y dejando la exportación a medias.

- experiment_fit_imm.m: Una vez seguidas las instrucciones (descargada la la BBDD de imágenes que sugiere) ejecuta correctamente.En la figura 3.14 se puede ver un ejemplo de los resultados obtenidos.

A pesar de utilizar versiones distintas del programa, se observa que los programas funcionan de manera similar, fallando por los mismos errores que aparecen cuando se ejecuta en el SO GNU/linux y obteniendo los mismos resultados cuando lo hacen correctamente.

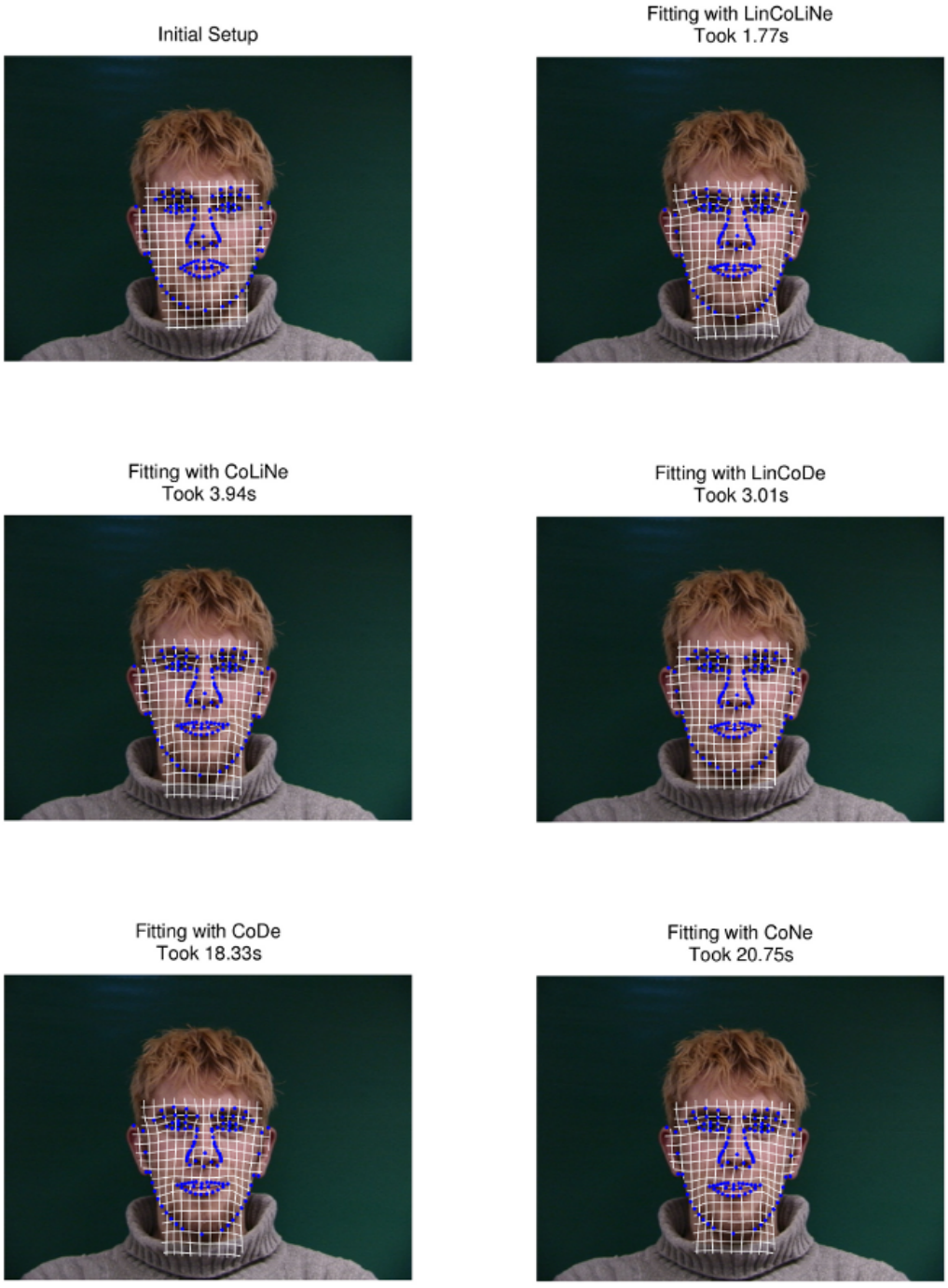


Figura 3.14: Resultado obtenido en windows del programa experiment_fit_imm.m sobre el usuario 1

3.4.2. Pruebas sobre el código en lenguaje C++

Preparamos un entorno de pruebas para la parte del código escrito en C++ que consiste en ejecutar un ejemplo de cada uno de los programas creados, que recordemos, utilizan la librería libaam_code, comparando los resultados obtenidos en cada SO. Para dicha comparación calcularemos la función md5 de los resultados.

3.4.2.1. Pruebas en GNU/Linux

En la figura 3.15 podemos observar la estructura creada. Explicamos sus componentes:

- datos: lugar donde guardamos los datos necesarios para la prueba.
  - aam_color_mixed_aam_all_192_912_60_1.code: AAM en formato compatible con la librería escrita en C++.
  - aam_color_mixed_aam_all_192_912_60_1.code.lm_fitter: fitter de dicho AAM.
  - berlusconi.lms: landmarks de la imagen de prueba.
  - berlusconi.ppm: imagen de prueba.
- resultado_aam-fitter-cli: lugar donde guardaremos los resultados del programa aam-fitter-cli.
- resultado_aam-inspector: lugar donde guardaremos los resultados del programa aam-inspector.
- prueba_aam-fitter-cli.sh: script de terminal que prueba el programa aam-fitter_cli.
- prueba_aam-inspector.sh: script de terminal que prueba el programa aam-inspector.
- prueba_completa.sh: script de terminal que prueba consecutivamente los programas aam-fitter-cli y aam-inspector.

Figura 3.15: Contenido de la carpeta de pruebas para el SO GNU/linux

Prueba del programa aam-fitter-cli

El programa aam-fitter-cli consiste en intentar ajustar, sobre una imagen de prueba, el modelo AAM siguiendo el método CoDe.

La ejecución del script de prueba_aam-fitter-cli.sh nos provoca la siguiente salida por pantalla:

```
**********************************************************************
                    Prueba del programa aam-fitter-cli
        Los resultados obtenidos estaran disponibles en la carpeta
                        resultado_aam-fitter-cli
**********************************************************************

Espere por favor, borrando resultados anteriores...
Borrado completado.
```

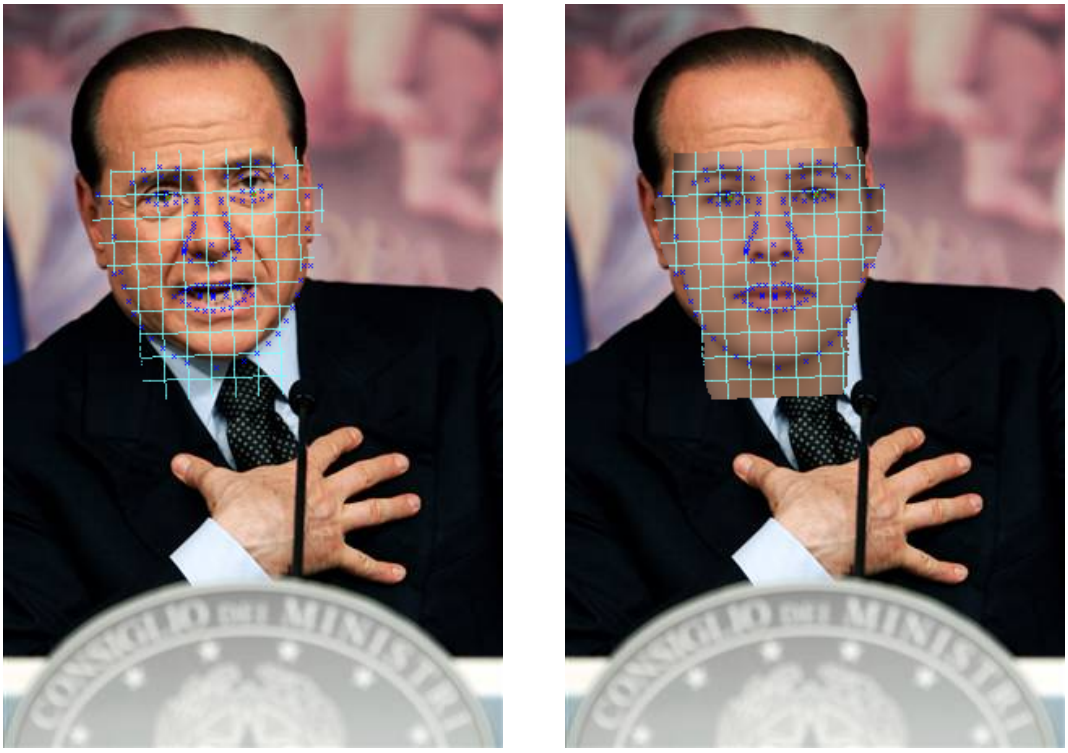
```
El comando que se va a ejecutar es:
aam-fitter-cli --aam=./datos/aam_color_mixed_aam_all_192_912_60_1.code --
   landmarks=./datos/berlusconi.lms --image=./datos/berlusconi.ppm --
   regularization-aam=50 --out ./resultado_aam-fitter-cli/

Loading AAM from ./datos/aam_color_mixed_aam_all_192_912_60_1.code
Loading landmarks from ./datos/berlusconi.lms
54.5 34.4  101 118.7
113.6 34.4  154 116.7
0 0  0 0
Loading image from ./datos/berlusconi.ppm
================================================================================
Landmark Regularization: 0.1
N_landmarks: 2
q: 64x1: [  -0.1032   0.0338 127.4755 165.8371   0.0000   0.0000   0.0000
    -0.0000   0.0000  -0.0000  -0.0000  -0.0000  -0.0000  -0.0000   0.0000
      0.0000   0.0000  -0.0000   0.0000   0.0000  -0.0000   0.0000
   0.0000   0.0000   0.0000   0.0000  -0.0000  -0.0000   0.0000  -0.0000
   -0.0000  -0.0000   0.0000  -0.0000  -0.0000  -0.0000  -0.0000   0.0000
     0.0000  -0.0000  -0.0000  -0.0000   0.0000   0.0000  -0.0000   0.0000
     -0.0000  -0.0000  -0.0000   0.0000   0.0000   0.0000  -0.0000
   0.0000  -0.0000  -0.0000   0.0000  -0.0000  -0.0000  -0.0000  -0.0000
     0.0000   0.0000   0.0000]
================================================================================
AAM Regularization:
 Steps: 3
 reg_from: 50.000000 to 5000.000000
 tex_from: 0 to 10

Smoothing: 3x1: [        4        2        0]
Reg:       3x1: [ 5000.0000 500.0000  50.0000]
Tex Reg:   3x1: [   0        0        0     ]
*.............ooooooooooooooooooooooooo#
*...............ooooooooooooooooooooooooo#
*..ooo........ooooooooooooooooooo#
q: 64x1: [  -0.1617   0.0965 129.4964 158.7782  -9.9285 370.2829 595.1899
   -284.3328 -334.1157 -419.9396 365.0740 -263.2918 -79.6391  53.2165
   99.6836  65.9247 -74.7874 -45.6557 -88.2848  22.6548   0.4528   9.1144
   -42.2370 -67.9173  -1.5196 -26.6138  19.3432  20.8086 -18.1212  -1.9471
     -7.0127 -23.5203  -7.9132  12.3202  13.7573 -25.1914   7.6365
   9.0744  16.4335  -2.8130  24.7013  15.7398  25.3849  -2.9474   8.0048
     4.3517  -1.7791  -4.7909 -13.2113   5.0730 -10.5524  -9.0073  -2.8531
      2.0968   0.2281   0.5590   1.6952  -1.3625  -3.7123   2.4692
   -1.3554   1.3329   2.1149   2.8618]

Calculando md5 del resultado. Espere por favor...
unix2dos: convirtiendo archivo md5_aam-fitter-cli.txt a formato DOS ...
c2dbd828b84836ba5f7716776d6a1a37  md5_aam-fitter-cli.txt
********************************************************************************
                          Ejecucion finalizada
                Compruebe la carpeta resultado_aam-fitter-cli
********************************************************************************
```

Es importante fijarse en las últimas líneas, donde aparece el resultado de la función md5 sobre las imágenes creadas por el programa, que se han guardado en la carpeta resultado_aam-fitter-cli. Estas se pueden ver en las figuras 3.16 y 3.17

Prueba del programa aam-inspector

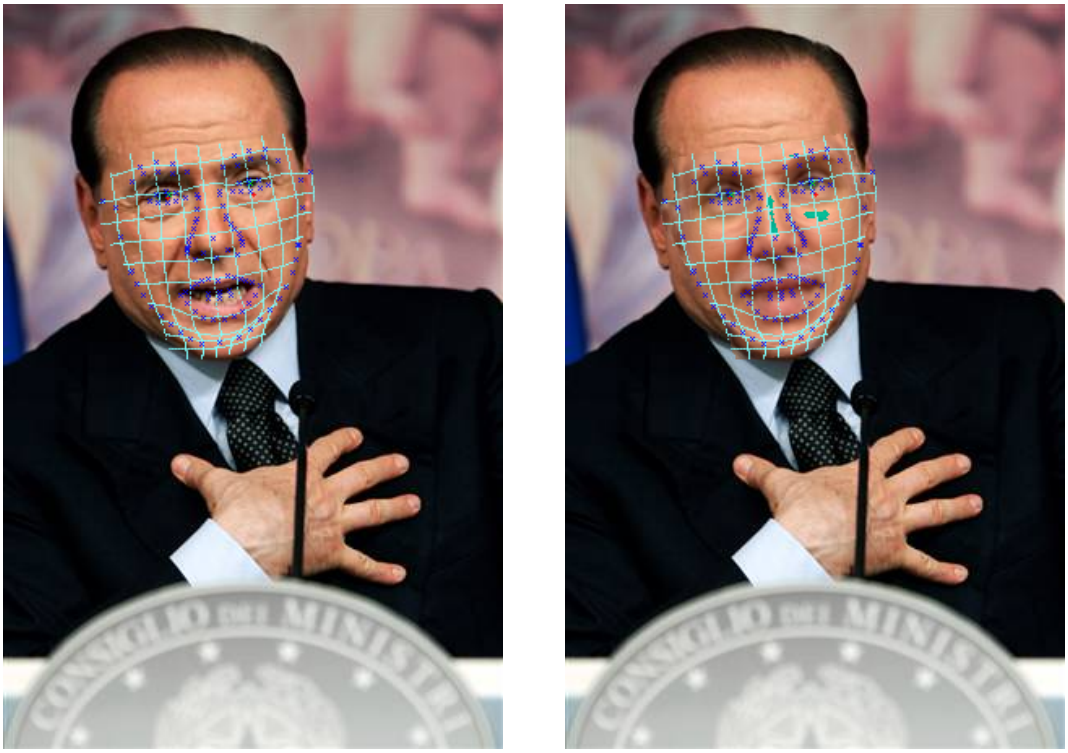
El programa aam-inspector realiza una exportación del contenido de la AAM a formato html para

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa



(a) Malla (b) Cara sintética

Figura 3.16: Posicionamiento de AAM sin ajustar



(a) Malla (b) Cara sintética

Figura 3.17: Ajuste de AAM siguiendo el método CoDe

ver los ajustes que lo forman mediante un navegador. Sería el equivalente al programa escrito en lenguaje MATLAB® experiment_display_aam.m.

La ejecución del script de prueba_aam-inspector.sh nos provoca la siguiente salida por pantalla:

```
**************************************************************************
                    Prueba del programa aam-inspector
        Los resultados obtenidos estaran disponibles en la carpeta
                          resultado_aam-inspector
**************************************************************************

Espere por favor, borrando resultados anteriores...
Borrado completado.

El comando que se va a ejecutar es:
aam-inspector ./datos/aam_color_mixed_aam_all_192_912_60_1.code --out ./
   resultado_aam-inspector/aam

Loading AAM from ./datos/aam_color_mixed_aam_all_192_912_60_1.code
Loading LM_FITTER from ./datos/aam_color_mixed_aam_all_192_912_60_1.code.
   lm_fitter

Calculando md5 del resultado. Espere por favor...
unix2dos: convirtiendo archivo md5_aam-inspector.txt a formato DOS ...
5cc6de9a8b9bd4a546b93f39bee888d3  md5_aam-inspector.txt
**************************************************************************
                            Ejecucion finalizada
                Compruebe la carpeta resultado_aam-inspector
**************************************************************************
```

Como el caso anterior, es importante fijarse en las últimas líneas, donde aparece el resultado de la función md5 sobre el resultado del programa, que se ha guardado en la carpeta resultado_aam-inspector.

3.4.2.2. Pruebas en Windows

En la figura 3.18 podemos observar la estructura creada para realizar las pruebas en este SO. Explicamos los componentes que difieren entre las versiones:

- datos: lugar donde guardamos los datos necesarios para la prueba. Mismo contenido que en GNU/linux.
- resultado_aam-fitter-cli: lugar donde guardaremos los resultados del programa aam-fitter-cli.exe.
- resultado_aam-inspector: lugar donde guardaremos los resultados del programa aam-inspector.exe.
- dll: lugar donde están las distintas DLL necesarias para ejecutar los programas compilados. Ver apartado A.2.3.
  - libblas.dll: librería BLAS
  - libfftw3-3.dll: librería FFTW3.
  - libfftw3f-3.dll: librería FFTW3.
  - libfftw3l-3.dll: librería FFTW3.
  - libgcc_s_dw2-1.dll: librería GNU.
  - libgcc_s_seh-1.dll: librería GNU.
  - libgcc_s_sjlj-1.dll: librería GNU.

- libgfortran-3.dll: librería GNU.
- liblapack.dll: librería BLAS.
- libquadmath-0.dll: librería GNU.
- libwinpthread-1.dll: librería GNU.
- md5.exe: programa que calcula la suma md5 de los ficheros requeridos.

- prueba_aam-fitter-cli.exe: programa aam-fitter-cli.exe.
- prueba_aam-inspector.exe: programa aam-inspector.exe
- prueba_aam-fitter-cli.bat: fichero BAT que prueba el programa aam-fitter-cli.exe.
- prueba_aam-inspector.bat: fichero BAT que prueba el programa aam-inspector.exe
- prueba_completa.bat: fichero BAT que prueba consecutivamente los programas aam-fitter-cli.exe y aam-inspector.exe.

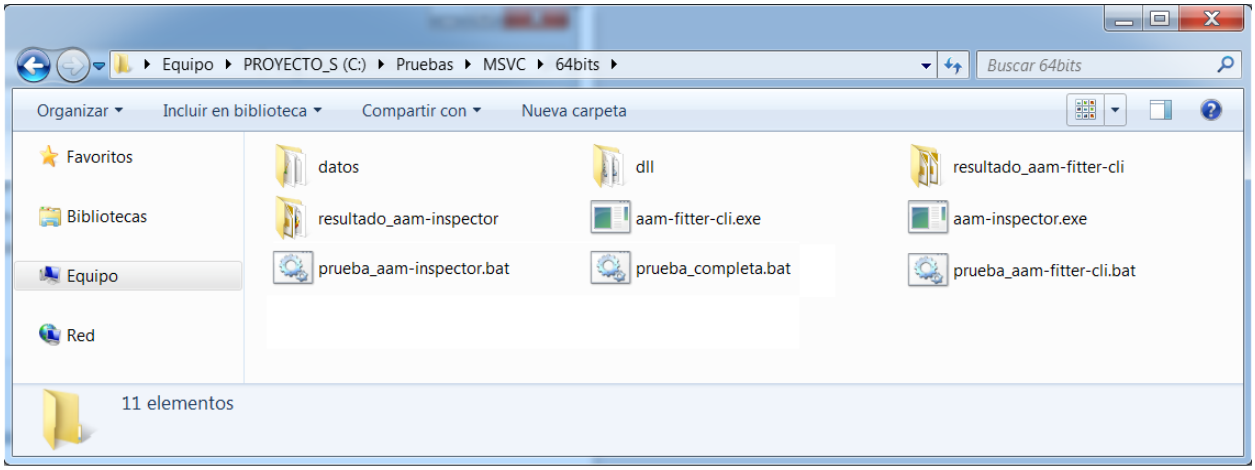


Figura 3.18: Contenido de la carpeta de pruebas para el SO Windows 7 Professional

Prueba del programa aam-fitter-cli.exe

La ejecución del fichero prueba_aam-fitter-cli.bat nos provoca la siguiente salida por pantalla:

```
**********************************************************************
                Prueba del programa aam-fitter-cli
       Los resultados obtenidos estaran disponibles en la carpeta
                    resultado_aam-fitter-cli
**********************************************************************
Espere por favor, borrando resultados anteriores...
Borrado completado.

El comando que se va a ejecutar es:
aam-fitter-cli --aam=.\datos\aam_color_mixed_aam_all_192_912_60_1.code --
   landmarks=.\datos\berlusconi.lms  --image=.\dat
os\berlusconi.ppm --regularization-aam=50 --out .\resultado_aam-fitter-cli
   \

Loading AAM from .\datos\aam_color_mixed_aam_all_192_912_60_1.code
```

```
Loading landmarks from .\datos\berlusconi.lms
54.5 34.4  101 118.7
113.6 34.4  154 116.7
0 0  0 0
Loading image from .\datos\berlusconi.ppm
================================================================================
Landmark Regularization: 0.1
N_landmarks: 2
q: 64x1: [  -0.1032   0.0338 127.4755 165.8371   0.0000   0.0000   0.0000
    -0.0000   0.0000  -0.0000  -0.0000  -0.0000
-0.0000  -0.0000   0.0000   0.0000   0.0000  -0.0000   0.0000   0.0000
    -0.0000   0.0000   0.0000   0.0000   0.0000   0.
0000  -0.0000  -0.0000   0.0000  -0.0000  -0.0000  -0.0000   0.0000
    -0.0000  -0.0000  -0.0000  -0.0000   0.0000   0.000
0  -0.0000  -0.0000  -0.0000   0.0000   0.0000  -0.0000   0.0000  -0.0000
    -0.0000  -0.0000   0.0000   0.0000   0.0000
-0.0000   0.0000  -0.0000  -0.0000   0.0000  -0.0000  -0.0000  -0.0000
    -0.0000   0.0000   0.0000   0.0000]
================================================================================
AAM Regularization:
 Steps: 3
 reg_from: 50.000000 to 5000.000000
 tex_from: 0 to 10

Smoothing: 3x1: [        4        2        0]
Reg:       3x1: [ 5000.0000 500.0000  50.0000]
Tex Reg:   3x1: [   0        0        0     ]
*.............ooooooooooooooooooooooooo#
*................ooooooooooooooooooooooooo#
*..ooo........ooooooooooooooooooo#
q: 64x1: [  -0.1617   0.0965 129.4964 158.7782  -9.9285 370.2829 595.1899
    -284.3328 -334.1157 -419.9396 365.0740 -263.29
18 -79.6391  53.2165  99.6836  65.9247 -74.7874 -45.6557 -88.2848  22.6548
      0.4528   9.1144 -42.2370 -67.9173  -1.5196
-26.6138  19.3432  20.8086 -18.1212  -1.9471  -7.0127 -23.5203  -7.9132
    12.3202  13.7573 -25.1914   7.6365   9.0744  16
.4335  -2.8130  24.7013  15.7398  25.3849  -2.9474   8.0048   4.3517
    -1.7791  -4.7909 -13.2113   5.0730 -10.5524  -9.00
73  -2.8531   2.0968   0.2281   0.5590   1.6952  -1.3625  -3.7123   2.4692
     -1.3554   1.3329   2.1149   2.8618]

Calculando md5 del resultado. Espere por favor...
c2dbd828b84836ba5f7716776d6a1a37
********************************************************************************
                        Ejecucion finalizada
             Compruebe la carpeta resultado_aam-fitter-cli
********************************************************************************
```

Observamos que obtenemos el mismo calculo md5, por lo que los resultados obtenidos en ambos Sistemas Operativos son idénticos, aunque el tiempo de ejecución es más elevado en Windows (14,5 frente a 10,4 segundos).

Prueba del programa aam-inspector.exe

La ejecución del fichero prueba_aam-inspector.bat nos provoca la siguiente salida por pantalla:

```
********************************************************************************
                    Prueba del programa aam-inspector
       Los resultados obtenidos estaran disponibles en la carpeta
                        resultado_aam-inspector
********************************************************************************
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

```
Espere por favor, borrando resultados anteriores...
Borrado completado.

El comando que se va a ejecutar es:
aam-inspector .\datos\aam_color_mixed_aam_all_192_912_60_1.code --out .\
    resultado_aam-inspector\aam

Loading AAM from .\datos\aam_color_mixed_aam_all_192_912_60_1.code
Loading LM_FITTER from .\datos\aam_color_mixed_aam_all_192_912_60_1.code.
    lm_fitter

Calculando md5 del resultado. Espere por favor...
5cc6de9a8b9bd4a546b93f39bee888d3
**************************************************************************
                          Ejecucion finalizada
                Compruebe la carpeta resultado_aam-inspector
**************************************************************************
```

Observamos que obtenemos el mismo calculo md5, por lo que los resultados obtenidos en ambos Sistemas Operativos son idénticos, aunque el tiempo de ejecución es más elevado en Windows (25,3 frente a 19,7 segundos).

Capítulo 4

Adaptación software MATLAB® a la base de datos GI4E

Vamos a explicar como hemos adaptado el software a nuestras necesidades, reutilizando parte de las funciones ya creadas en el software original, modificando otras y escribiendo algunas nuevas.

4.1. Gestión de la base de datos GI4E

Una vez analizada la base de datos en el apartado 2.2, así como el software original, constatamos los siguientes inconvenientes:

- La base de datos GI4E ocupa 43,5 GB, cuando la parte que realmente necesitamos (12 imágenes en alta resolución por usuario, con sus correspondientes landmarks) ocupa menos de 5,5 GB.
- A cada imagen de usuario le deben acompañar otra de máscara con la situación de la cara (imagen_mask.png) y un fichero con los landmarks de esa imagen (imagen.png.landmarks). Sin embargo, los landmarks de todas las imágenes de un usuario están en un sólo fichero, que además posee un formato distinto al que utiliza el software original, y no existen las imágenes de máscara.
- Las imágenes de los usuarios tiene mucho espacio "desperdiciado": hay mucho fondo que es superfluo para nuestro propósito.

Por todo ello decidimos realizar las siguientes acciones:

- Implementar un método para importar de la base de datos GI4E sólo los recursos que necesitamos.
- Crear, a partir de los landmarks, una imagen de máscara para cada imagen de usuario. La máscara se crea en el canal de transparencia de un fichero PNG, siendo el elemento que no es trasparente de la misma.
- Los landmarks se van a separar, creando un fichero para cada imagen de usuario. Además se van a adaptar al formato utilizado por el software original (3 columnas: número de punto, coordenadas en el eje X y coordenadas en el eje Y).
- Se va a crear la posibilidad de trabajar con imágenes "recortadas", en las que, basándonos en los landmarks, se elimina gran parte del fondo superfluo. Este será el tipo de imagen de trabajo recomendada.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- Se va a implementar un procedimiento para poder eliminar todos los ficheros que se crean en los puntos anteriores, para poder dejar la base de datos con el número mínimo de datos necesarios.

Es muy importante preparar la base de datos (importar la base de datos GI4E, crear y exportar los ficheros y en caso de necesidad borrado de los mismos) antes de pasar a las siguientes fases.

4.1.1. Programas

Para la gestión de la base de datos desarrollamos 3 programas. Se pueden ver sus organigramas en la figura 4.1.

p_1a_BBDD_import.m
Programa encargado de importar los datos relevantes de la base de datos GI4E a la carpeta que especifiquemos. Se mantendrá la estructura de carpetas presente en la base de datos.
Valores modificables:

- DDBB_GI4E_path='D:\GI4E Database'; Ubicación de la BBDD GI4E
- DDBB_path='BBDD'; Ubicación de la BBDD

p_1b_BBDD_export_all.m
Programa encargado de crear, a partir de una imagen de usuario y sus correspondientes puntos de control, la imagen recortada, las máscaras de ambas imágenes y sus ficheros de landmarks asociados. Por ejemplo, después de su ejecución, de los ficheros:

- 003_01.png: imagen en alta resolución
- Usuario_003.txt: landmarks del usuario 003.

Surgen los siguientes ficheros adicionales:

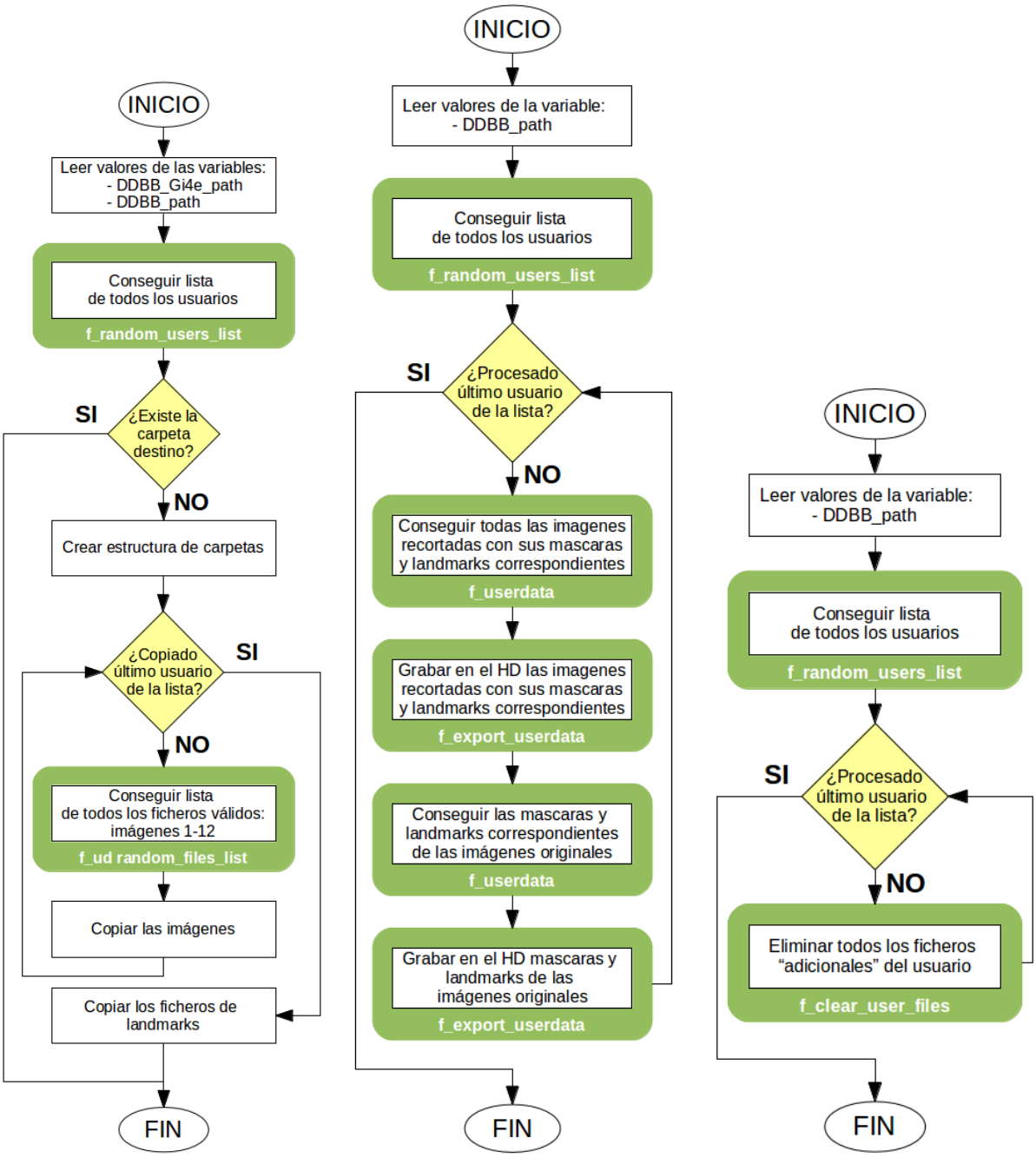
- 003_01_mask.png: máscara de la imagen en alta resolución.
- 003_01.png.landmarks: landmarks de la imagen en alta resolución.
- 003_01_crop.png.landmarks: landmarks de la imagen recortada.
- 003_01_crop.png: imagen recortada.
- 003_01_crop_mask.png: máscara de la imagen recortada.

Valores modificables:

- DDBB_path='BBDD'; Ubicación de la BBDD

p_1c_BBDD_clear.m
Programa encargado de eliminar los ficheros creados por el programa anterior, dejando en la base de datos sólo las imágenes originales en alta resolución y los ficheros de landmarks unificados por usuario.
Valores modificables:

- DDBB_path='BBDD'; Ubicación de la BBDD



(a) p_1a_BBDD_import.m (b) p_1b_BBDD_export_all.m (c) p_1c_BBDD_clear.m

Figura 4.1: Organigramas programas que gestionan la BBDD

4.1.2. Funciones

f_clear_user_files.m
Borrar los ficheros adicionales de un usuario.

f_display_userdata.m
Muestra en pantalla las imágenes contenidas en la estructura de datos userdata.

f_export_ud_images.m
Exporta una imagen al disco duro.

f_export_ud_landmarks.m

Exporta los datos de landmarks a un fichero del disco duro.

f_export_ud_mask.m
Exporta una imagen de máscara al disco duro.

f_export_userdata.m
Exporta los ficheros contenidos en la estructura de datos userdata al disco duro.

f_random_users_list.m
Devuelve listas aleatorias de usuarios.

f_ud_create_paths.m
Obtiene las rutas de los ficheros de usuario a utilizar.

f_ud_create_struct.m
Crea los ficheros adicionales de un usuario.

f_ud_images.m
Esta es la función principal para el tratamiento de los ficheros de imagen.

f_ud_landmarks.m
Esta es la función principal para el tratamiento de los datos de landmarks.

f_ud_landmarks_ajust.m
Ajusta los datos de landmarks para los ficheros de imagen recortados.

f_ud_landmarks_import.m
Importa los datos de landmarks de un usuario. Primero intenta importarlo de un fichero .landmarks y, en caso de que no existan, del fichero general de landmarks de ese usuario.

f_ud_landmarks_import_from_original_file.m
Devuelve los datos de landmarks de un usuario a partir del fichero general de landmarks de ese usuario.

f_ud_landmarks_mask.m
Transforma los landmarks para crear la imagen máscara.

f_ud_mask.m
Esta es la función principal para el tratamiento de los ficheros de imagen de máscara.

f_ud_random_files_list.m
Devuelve una lista aleatoria de ficheros de usuario.

f_userdata.m
Función principal para obtener todos los datos de un usuario (imágenes, máscaras y landmarks) y almacenarlos en la estructura de datos userdata.

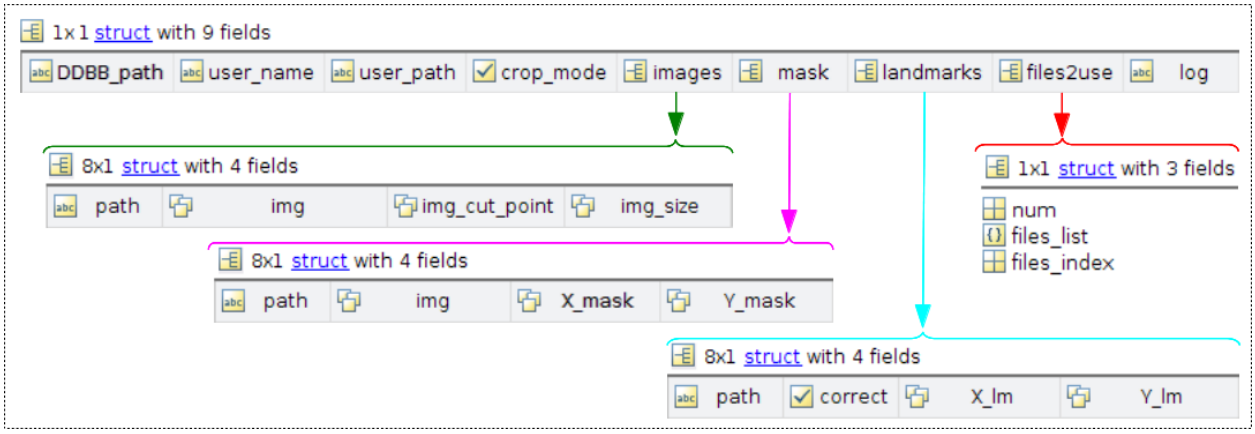


Figura 4.2: Organización interna userdata

4.1.3. Estructuras de datos

Para poder gestionar de manera más cómoda los datos de un usuario se ha creado una estructura que los contendrá.

mat userdata

Esta estructura de datos contiene los datos de las imágenes escogidas de un usuario. También se incluye información sobre los landmarks correspondientes a esas imágenes y las máscaras de las mismas. Su organización interna se puede ver en la figura 4.2.

Pasamos a explicar su contenido:

- BBDD_path: ruta de la base de datos.
- user_name: nombre del usuario al que corresponden los datos.
- user_path: ruta de los datos del usuario.
- crop_mode: valor booleano que indica si estamos utilizando imágenes recortadas o no.
- images: estructura que contiene la información de las imágenes del usuario. Tendrá tantos elementos como imágenes de usuario escogidas.
  - path: ruta de la imagen.
  - img: imagen propiamente dicha (3 matrices).
  - img_cut_point: si estamos utilizando imágenes recortadas, y la hemos tenido que recortar de la original (no la hemos importado del disco duro), en esta variable aparecerá el punto de origen del recorte.
  - img_size: tamaño de la imagen.
- mask: estructura que contiene la información de las imágenes de máscara. Tendrá tantos elementos como imágenes de usuario escogidas.
  - path: ruta de la imagen de máscara.
  - img: imagen de máscara propiamente dicha (3 matrices).
  - X_mask: vector, con 32 elementos, con los datos de los puntos en el eje X necesarios para obtener la imagen de máscara.
  - Y_mask: vector, con 32 elementos, con los datos de los puntos en el eje Y necesarios para obtener la imagen de máscara.
- landmarks: estructura que contiene la información de los landmarks. Tendrá tantos elementos como imágenes de usuario escogidas.

  - path: ruta del fichero de landmarks.
  - correct: valor booleano que indica si los datos de los landmarks están comprobados (necesario cuando utilizamos imágenes recortadas).
  - X_lm: vector, con 52 elementos, con los datos de los puntos en el eje X.
  - Y_lm: vector, con 52 elementos, con los datos de los puntos en el eje Y.
- files2use: estructura que contiene la información de cuáles de las 12 imágenes de usuario se han escogido.
  - num: número de imágenes escogidas.
  - files_list: nombres de los ficheros a escoger, siguiendo la estructura nombre de usuario_+numero del fichero+_crop_mode
  - files_index: listado de los números de los ficheros a utilizar. Se obtienen de manera aleatoria.
- log: contiene información sobre que imágenes se han escogido y el proceso seguido para obtenerlas.

4.2. Fase de entrenamiento

En la fase de entrenamiento tenemos que desarrollar los programas encargados de crear un AAM con las imágenes de la base de datos GI4E. Deberemos poder especificar el número de usuarios con los que lo crearemos así como cuantas imágenes, de las 12 disponibles, serán utilizadas. Tanto los usuarios como los ficheros se deben escoger de manera aleatoria.
Para el estudio hemos decido que crearemos 36 AAM variando los parámetros disponibles. Evidentemente usaremos siempre las mismas imágenes. Se usarán 8 imágenes por usuario y se escogerán 60 usuarios, quedando el resto de usuarios disponibles para las siguientes fases.

4.2.1. Programas

En este caso se va a hacer todo en un único programa, que se encargará de todos los requisitos.

p_2_training.m
Está basado en el programa original build_models.m, al que hemos realizado modificaciones para adaptarlo a nuestras necesidades.
Valores modificables referentes a la base de datos:

- DDBB_path='BBDD'; Ubicación de la BBDD
- num_images=8; Imágenes por usuario
- num_users_training=60; Usuarios de entrenamiento
- crop_images=true; Uso de imágenes recortadas
- export_DDBB_files=false; Exporta ficheros adicionales del usuario
- clear_DDBB_files=false; Borra ficheros adicionales del usuario
- clear_DDBB_full=false; Borra todos los ficheros adicionales

Valores modificables referentes a las opciones de creación del AAM:

- model='mixed_aam'; Nombre del AAM (no cambia)
- train_set_name='all'; Nombre del AAM (no cambia)

- model_resolution = 192; Resolución del modelo (128,192,256)
- model_size=60; Tamaño del modelo (60,80,100)
- use_color=true; Uso de color (siempre activo)
- blur=1; Uso de desenfoque (0,1)
- use_symmetrie=false; Uso de simetría (0,1)

Valores modificables referentes a las opciones de creación del AAM:

- aam_path='aam'; Ubicación de los AAM generados
- create_log=true; Crea fichero log
- display_result=false; Muestra AAM generado
- export_aam2code=false; Exporta AAM a formato librería C++
- export_aam2html=false; Exporta AAM a html para visualización
- hack_program=false; Genera el AAM con userdata existente

En la figura 4.4 se puede ver su organigrama.

4.2.2. Funciones

Al estar basado en el software original se utilizan una serie de funciones del mismo. En el anexo C se puede ver cual es su utilidad concreta. En este punto solamente vamos a enumerarlas. Funciones originales para creación del fichero AAM:

- aam_procrustes.m
- build_model.m
- covariance_svd.m
- gaussian_blur.m
- rbf_calculate_distance_matrix.m
- rbf_create_basis.m
- rbf_evaluate.m
- rbf_find_coefficients.m
- aam_prepare_compositional.m
- aam_prepare_compositional_original.m
- aam_prepare_warp_composition.m
- ba_extrapolate_gridded.m
- ba_interp2.mexw64
- vec.m

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Funciones originales para exportar el fichero AAM al formato de la librería creada en C++ (libaam_code):

- aam_prepare_code_fitter.m
- aam_save_code.mexw64
- aam_save_code_fitter.m

Funciones originales para exportar el fichero AAM a html:

- aam_prepare_warp_composition.m
- aam_render.m
- ba_affine.mexw64
- ba_rasterize.mexw64
- f_aam_to_html.m

Pasamos a explicar el resto de funciones utilizadas.

- f_display_aam.m
  Muestra en pantalla el contenido del AAM generado.
- f_export_aam2code.m
  Exporta el AAM generado al formato de la librería creada en C++ (libaam_code). Basado en el programa original example_prepare_aam_for_libaam_code.m
- f_export_aam2html.m
  Exporta el AAM generado a html para poder visualizar su contenido en un navegador web. Basado en el programa original experiment_display_aam.m
- f_log_create.m
  Crea un fichero de log del proceso de entrenamiento. Contiene la lista de los usuarios de entrenamiento, de optimización y de testeo, así como la lista de las imágenes concretas de cada usuario utilizadas.

4.2.3. Estructuras de datos

En este apartado se genera una estructura de datos que contiene un implementación de AAM basada en el software original.

- aam
  En nuestro caso añadimos unos campos en el que se guarda información sobre el modo en el que se ha creado. Se puede ver su contenido en la figura 4.3.
  - model_name: nombre completo del modelo.
  - UPT: número de usuarios utilizados para su creación (users_training).
  - IPU: número de imágenes por usuario.
  - list_users_training: listado de nombres de los usuarios de entrenamiento utilizados.

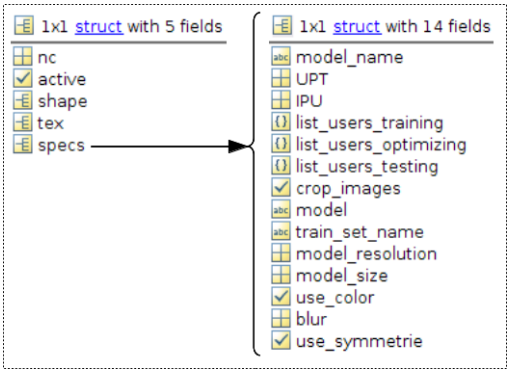


Figura 4.3: Estructura interna AAM

- list_users_optimimizing: listado de nombres de los usuarios de escogidos para la fase de optimización.
- list_users_testing: listado de nombres de los usuarios de escogidos para la fase de testeo.
- crop_images: valor booleano que indica si estamos utilizando imágenes recortadas o no.
- model: nombre del modelo.
- train_set_name: nombre del modelo.
- model_resolution: resolución del modelo.
- model_size: tamaño del modelo.
- use_color: variable booleana que indica si utilizamos color o no.
- blur: variable que indica si utilizamos desenfoque o no.
- use_symmetrie: variable booleana que indica si utilizamos simetría o no.

4.2.3.1. Lista AAM generados

Los nombres de los AAM generados nos desvelan las especificaciones con la que se han creado:

aam_color_mixed_aam_all_Resolution_Num-imágenes_Size_Blur_(Num-usersxImg-per-user).mat

En la tabla 4.1 se puede ver la relación de los 36 AAM generados en la fase de entrenamiento. En el Anexo D se puede ver un ejemplo de un fichero log generado en esta fase. En él se puede observar que usuarios se han utilizado para su creación, que imágenes concretas se han escogido de cada usuario, el tamaño del modelo, la resolución del mismo...

	AAM (.mat file)	Resolution	Simetrie	Size	Blur
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat	128	false	60	false
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat	128	false	60	true
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	128	false	80	false
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	128	false	80	true
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	128	false	100	false
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	128	false	100	true
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat	128	true	60	false
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat	128	true	60	true
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	128	true	80	false
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	128	true	80	true
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	128	true	100	false
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	128	true	100	true
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat	192	false	60	false
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat	192	false	60	true
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	192	false	80	false
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	192	false	80	true
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	192	false	100	false
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	192	false	100	true
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	192	true	60	false
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	192	true	60	true
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	192	true	80	false
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	192	true	80	true
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	192	true	100	false
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	192	true	100	true
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	256	false	60	false
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat	256	false	60	true
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	256	false	80	false
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	256	false	80	true
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	256	false	100	false
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	256	false	100	true
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat	256	true	60	false
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat	256	true	60	true
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	256	true	80	false
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	256	true	80	true
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	256	true	100	false
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	256	true	100	true

Tabla 4.1: Lista de AAM creados en la fase de entrenamiento

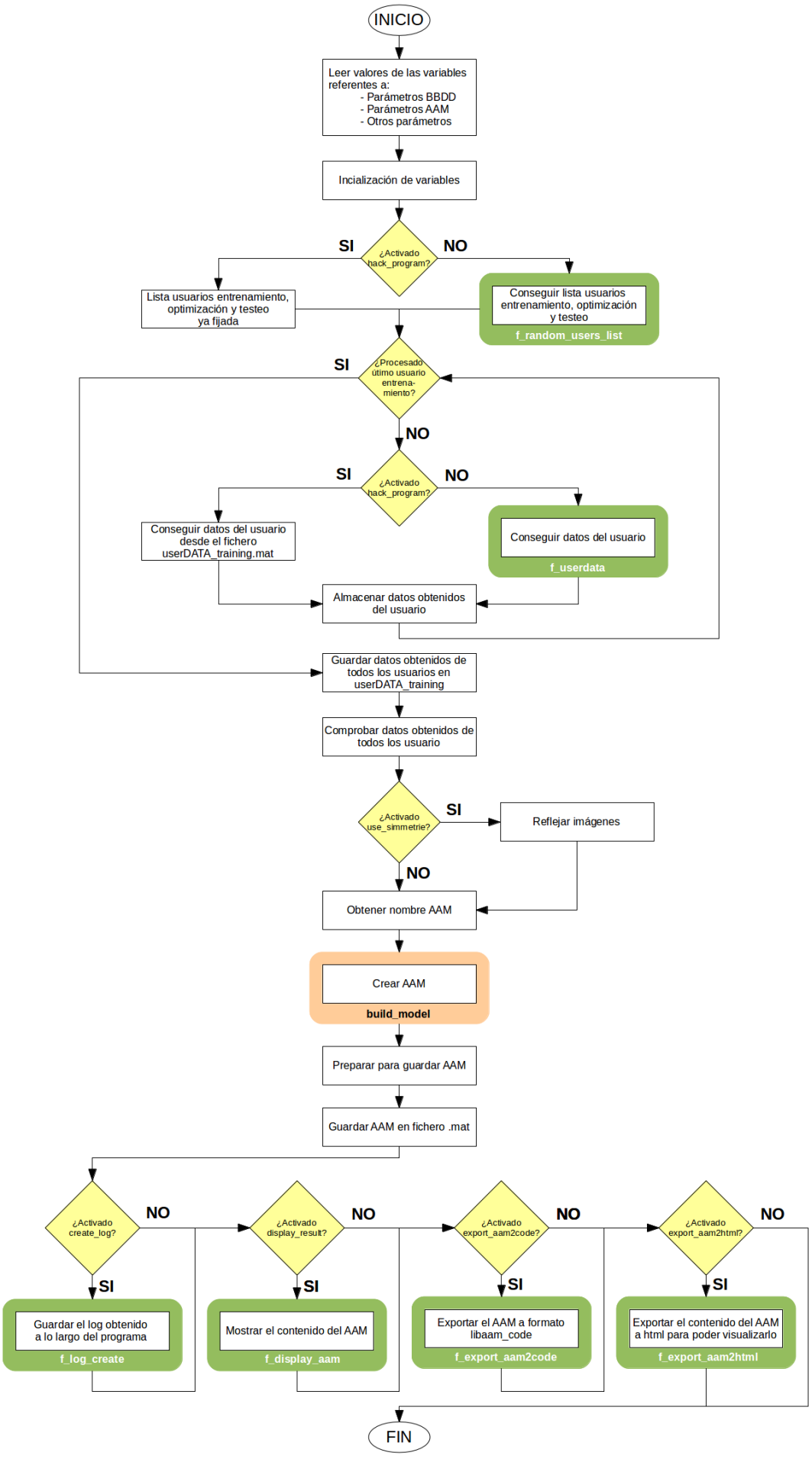


Figura 4.4: Organigrama p_2_training

4.3. Fase de optimización

En la fase de optimización tenemos que desarrollar los programas encargados de aplicar los AAM creados en la fase de entrenamiento sobre la lista de usuarios de optimización (recordemos que la lista de usuarios está en el interior del propio AAM).
Tendremos que aplicar los 36 AAM a cada usuario (20 usuarios en total), almacenar los resultados obtenidos (para ello crearemos una estructura de datos llamada ImagesDATA) y representarlos para su análisis.

4.3.1. Programas

Hemos tenido que realizar tres programas, uno para la creación de los datos y dos para su representación.

p_3_optimizing.m
Se encarga de aplicar los 36 AAM disponibles a los usuarios de optimización. Guardará los resultados obtenidos en una estructura ImagesDATA. Habrá 36 ficheros ImagesDATA por cada usuario de la fase de optimización, 720 en total.
Valores modificables para su uso:

- aam_path='aam'; Ubicación de los AAM a testear
- DDBB_path='BBDD'; Ubicación de la BBDD
- crop_images=true; Uso de imágenes recortadas
- display_fit=false; Muestra imágenes del proceso
- result_path='resultados_optimizing'; Ubicación de los resultados

Se puede ver el organigrama del programa en la figura 4.5a.

p_4_graphics_optimizing.m
Se encarga de leer los datos de todos los ficheros ImagesDATA y representarlos en 5 gráficos distintos, uno por cada método. En cada gráfico habrá 36 diagramas de caja, uno para cada AAM. El resultado puede verse en las figuras 5.2, 5.3, 5.4, 5.5 y 5.6.
Valores modificables para su uso:

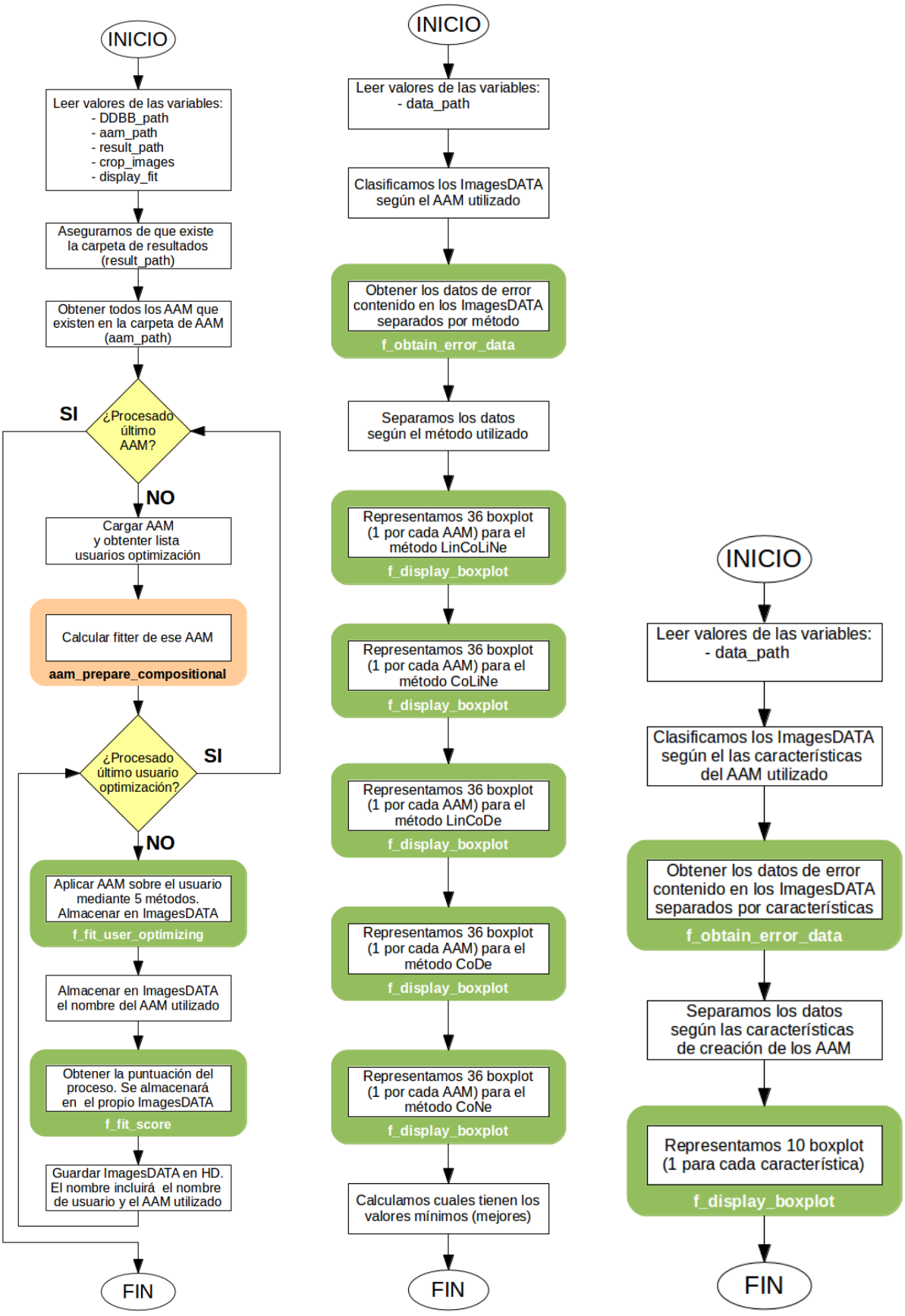
- data_path='resultados_optimizing'; Ubicación de los resultados

Se puede ver su organigrama en la figura 4.5b.

p_4_graphics_optimizing_all.m
Se encarga de leer los datos de todos los ficheros ImagesDATA y representarlos en un gráfico, agrupando los datos en 10 diagramas de caja, uno por cada parámetro. El resultado puede verse en la figura 5.1.
Valores modificables para su uso:

- data_path='resultados_optimizing'; Ubicación de los resultados

Se puede ver su organigrama en la figura 4.5c.



(a) p_3_optimizing.m (b) p_4_graphics_optimizing (c) p_4_graphics_optimizing_all

Figura 4.5: Organigramas programas fase optimización

4.3.2. Funciones

Al estar basado en el software original se utilizan una serie de funciones del mismo. En el anexo C se puede ver cual es su utilidad concreta. En este punto solamente vamos a enumerarlas.
Funciones originales para aplicar el AAM en las imágenes de los usuarios:

- aam_fc_gradient.m
- aam_fc_linear_gradient.m
- aam_fit_code.m
- aam_fit_code_mex.mexw64
- aam_fit_coline.m
- aam_fit_cone.m
- aam_fit_lincode.m
- aam_fit_lincoline.m
- aam_fit_regularized.m
- aam_fit_to_warp.m
- aam_fit_to_warp_mex.mexw64
- aam_gradient_fc_mex.mexw64
- aam_ic_gradient.m
- ba_affine.mexw64
- has_java.m

Resto de funciones utilizadas para la aplicación de los AAM en las imágenes de los usuarios:

- f_fit_display_warp.m
  Devuelve los puntos calculados (landmarks_fit) de una imagen y los representa encima de ella. Basada en la función original aam_display_warp.m.

- f_fit_image.m
  Devuelve los puntos calculados (landmarks_fit) de una imagen, el tiempo que ha costado calcularlos y el error que presenta respecto a los puntos de referencia marcados de manera manual. Basada en el programa original experiment_fit_imm.m

- f_fit_load_parameters.m
  Devuelve los parámetros de posicionamiento iniciales. En esta función se especifica los puntos donde se va a posicionar el AAM para proceder a su ajuste. Se utilizan los puntos 9, 17, 34 y 38 (partes exteriores de boca y ojos). Basada en la función original aam_load_image_and_parameters_from_imm.m.

f_fit_score.m
Agrega a la estructura de datos ImagesDATA un resumen de los errores que presentan las imágenes respecto a los puntos de referencia marcados manualmente. Puede servirnos para hacernos una idea de la calidad del ajuste.

f_fit_user_optimizing.m
Crea una estructura de datos ImagesDATA con los datos de de todas las imágenes de usuario a las que se les ha aplicado todos los algoritmos de ajuste.

Funciones utilizadas para la representación de los datos:

f_display_best_image.m
Representa la imagen mejor ajustada de un usuario, sin importar el método que se haya utilizado.

f_display_best_image_per_method.m
Representa las 5 mejores imágenes de un usuario, una por cada método de ajuste.

f_display_boxplot.m
Muestra en pantalla el diagrama de caja y devuelve los datos más significativos del mismo (valor mínimo, percentil 25, percentil 50 (mediana), percentil 75 y valor máximo).

f_obtain_error_data.m
Devuelve todos los datos de los errores de ajuste que contiene la estructura de datos ImagesDATA. Los devuelve clasificados por método utilizado y sin clasificar.

4.3.3. Estructuras de datos

En este apartado se genera una estructura de datos en el que vamos almacenando los resultados obtenidos de aplicar el AAM sobre las imágenes de un usuario.

ImagesDATA
En esta estructura almacenamos el resultado de aplicar los distintos métodos implementados sobre todas las imágenes de un usuario. En la figura 4.6 se puede ver su organización interna.

- user_path: ruta de los datos del usuario.
- user_name: nombre del usuario.
- images: estructura donde almacenamos la información de las imágenes. Tendrá 12 elementos distintos, ya que verificamos todas las imágenes de un usuario.
  - img_path: ruta de la imagen.
  - landmark_path: ruta del fichero landmark.
  - T_landmarks: tabla con los 52 puntos de referencia marcados a mano, así como los calculados aplicando cada método.
  - T_time: tabla con el tiempo que ha costado aplicar cada método sobre esa imagen.
  - T_error: tabla con la distancia euclídea entre el punto calculado con ese método y el punto marcado de manera manual.
  - T_error_med: tabla con la distancia euclídea media para esa imagen por método.
- aam_used: nombre del AAM utilizado.

- score_per_method: tabla con la puntuación de cada método (media del error medio de las 12 imágenes).
- score: media de la puntuación de los cinco métodos.

Los distintos ImagesDATA generados en esta fase, 36 por cada uno de los 20 usuarios, se nombran siguiendo el patrón:

ImagesDATA_+nombre de usuario +nombre AAM utilizado

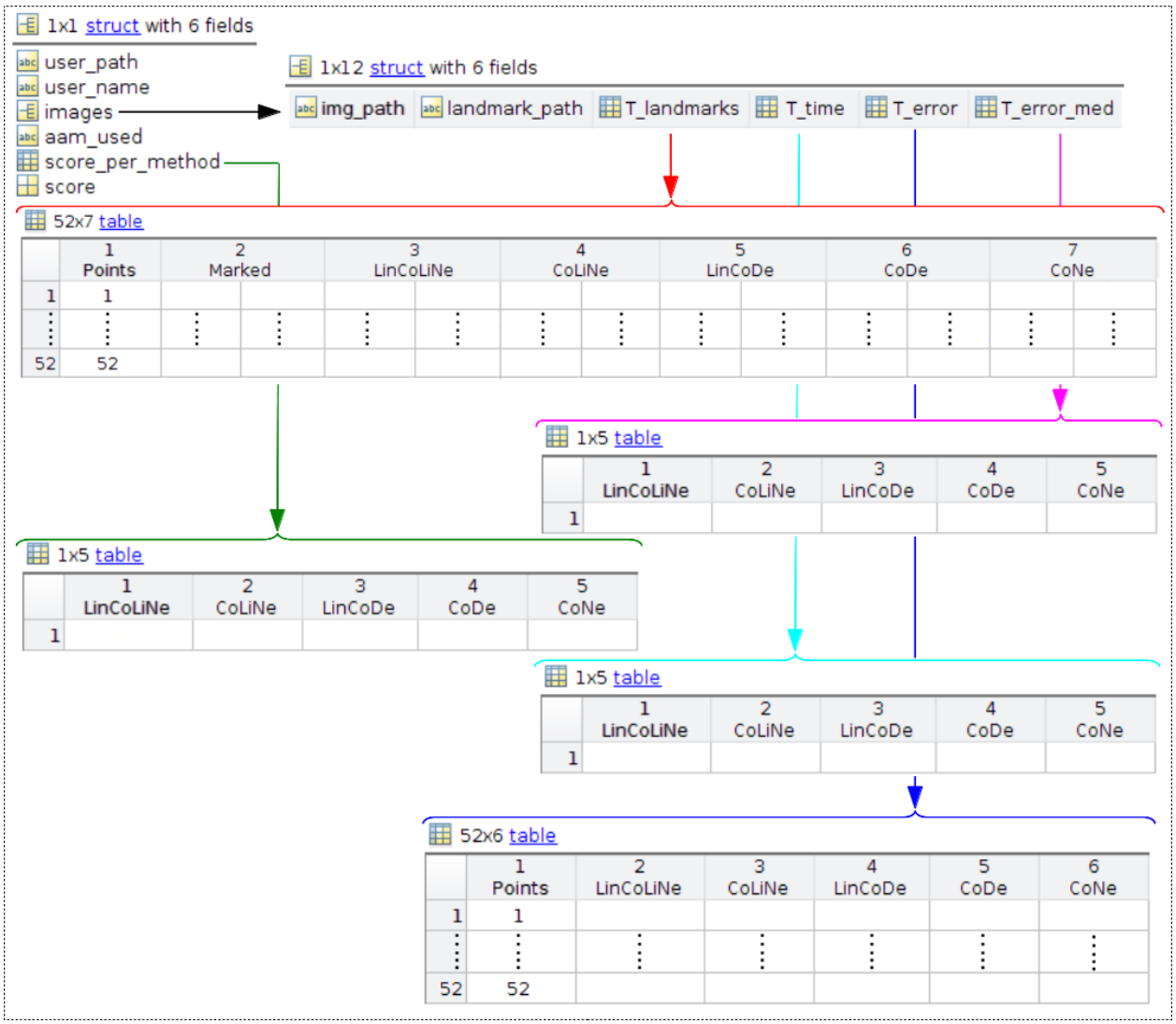


Figura 4.6: Organización interna ImagesDATA fase optimización

4.4. Fase de testeo

Esta fase es muy parecida a la anterior, pero en ésta sólo tenemos que probar los 5 AAM óptimos (uno por cada método) en los 23 usuarios de testeo. Además debemos conseguir los datos de uso de memoria.

4.4.1. Programas

Hemos tenido que realizar tres programas, uno para la creación de datos y dos para su representación y análisis.

p_5_testing.m
Se encarga de probar en cada usuario los 5 AAM óptimos. El programa es similar al utilizado en la fase de optimización, pero aquí calculamos un método en todos los usuarios antes de pasar al siguiente método, en vez de calcular todos los métodos en un usuario antes de pasar al siguiente usuario.
Valores modificables para su uso:

- aam_path='aam'; Ubicación de los AAM a testear
- DDBB_path='BBDD'; Ubicación de la BBDD
- crop_images=true; Uso de imágenes recortadas
- display_fit=false; Muestra imágenes del proceso
- result_path='resultados_testing'; Ubicación de los resultados
- debug_memory=true; Guarda información de memoria

En la figura 4.7 se puede ver su organigrama. Como el uso del debug para obtener la memoria utilizada (la información se guarda en la estructura profileDATA) ralentiza el proceso calcularemos dos veces, una para tomar tiempos y otra para tomar la medición de uso de memoria.

p_6_convert_mat2xls.m
Se encarga de exportar los datos de memoria usada y tiempo utilizado, que están dentro de la estructura de datos ImagesDATA, a un fichero excel. Para realizarlo utiliza el fichero template.xls y sólo se puede ejecutar en el SO Windows.
Valores modificables para su uso:

- input_path='resultados_testing'; Ubicación de los resultados de testeo

En la figura 4.8a se puede ver su organigrama.

p_6_graphics_testing.m
Se encarga de crear un un diagrama de caja por cada método de los datos contenidos en los ImagesDATA de los usuarios de testeo. El resultado puede verse en la figura 5.7.
Valores modificables para su uso:

- data_path='resultados_testing'; Ubicación de los resultados de testeo

En la figura 4.8b se puede ver su organigrama.

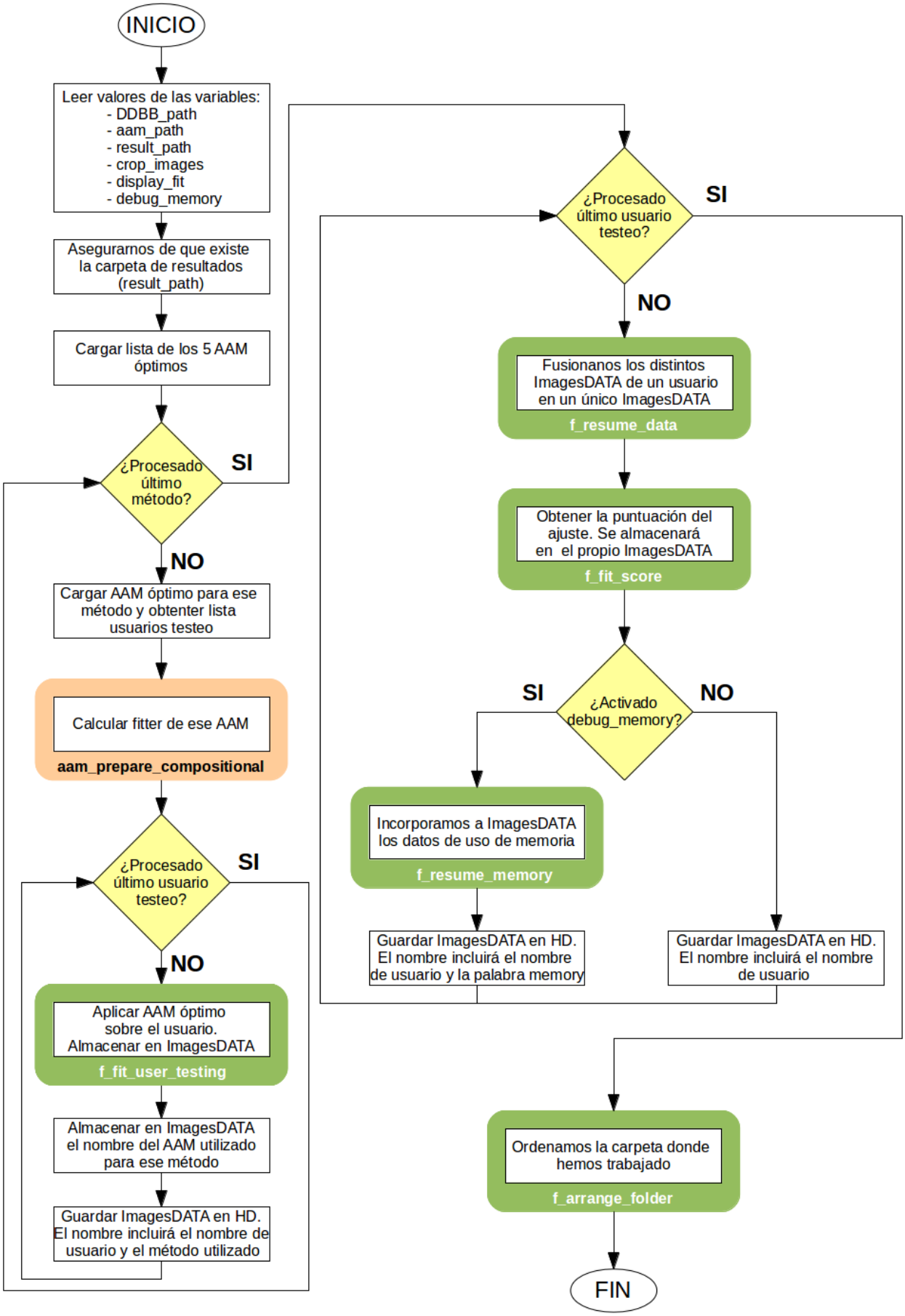
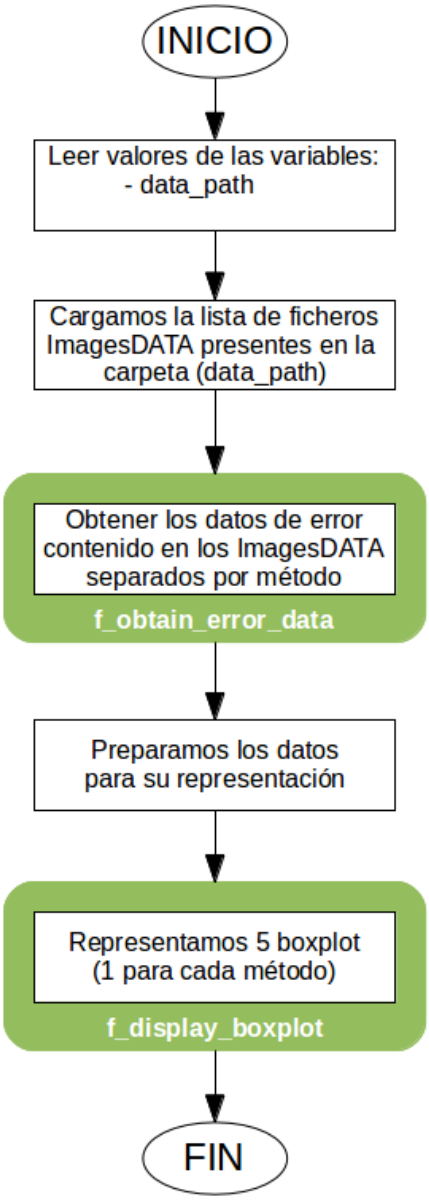
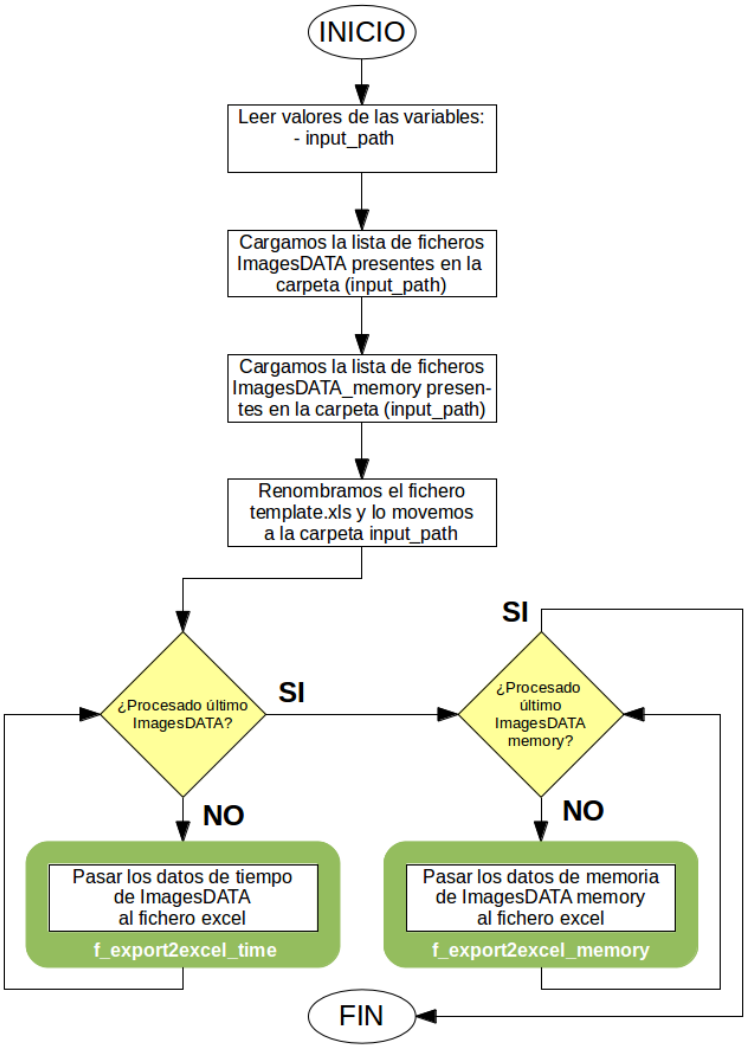


Figura 4.7: Organigrama p_5_testing.m



(a) p_6_convert_mat2xls.m



(b) p_6_graphics_testing.m

Figura 4.8: Organigramas programas representación fase testeo

4.4.2. Funciones

Al ser tan parecidos los programas a los de la fase anterior, comparten muchas funciones. Además de ellas, se utilizan las siguientes:

f_arrange_folder.m
Clasifica los ImagesDATA por método y los profileDATA según el usuario, guardándolos en sus respectivas carpetas.

f_fit_user_testing.m
Crea una estructura de datos ImagesDATA con los datos de todas las imágenes de usuario a las que se les ha aplicado únicamente un método.

f_resume_data.m

Fusiona todos los datos de los ImagesDATA de un usuario, que están separados según método, en un único ImagesDATA.

f_resume_memory.m
Agrega a la estructura de datos ImagesDATA los datos de uso de memoria presentes en los ficheros profileDATA.

f_resume_memory_mex.m
Agrega a la estructura de datos ImagesDATA los datos de uso de memoria presentes en los ficheros profileDATA, cuando se ha utilizado funciones MEX en el proceso, ya que las asignaciones de memoria aparecen en distinto lugar.

Funciones utilizadas para la exportación de los datos:

f_export2excel_memory.m
Exporta a un fichero excel los datos de memoria utilizada que se encuentran en la estructura de datos ImageDATA.

f_export2excel_time.m
Exporta a un fichero excel los datos de tiempo utilizado que se encuentran en la estructura de datos ImageDATA.

4.4.3. Estructuras de datos

En este apartado se utiliza una versión ligeramente modificada de ImagesDATA y la estructura profileDATA que contiene toda la información del uso de memoria del proceso.

ImagesDATA (versión fase testeo)
En esta fase la estructura recibe un par de modificaciones debido al hecho de que cada usuario de la fase de testeo es analizado con un AAM distinto dependiendo del método de ajuste utilizado.
En primer lugar se crea una estructura ImagesDATA que sólo contiene los puntos calculados para ese método y el tiempo que ha costado hacerlo. Lo guardamos con un nombre que sigue el siguiente patrón:

ImagesDATA_+nombre de usuario_+método de ajuste utilizado

Una vez hemos calculado todos los métodos para todos los usuarios, fusionamos todos los datos en un único ImagesDATA por usuario. Además hemos de incluir los datos de uso de memoria en caso de que así lo hubiésemos configurado. En la figura 4.9 pueden verse con más detalle los añadidos que ahora comentamos:

- images: sufre varias adiciones.
  - T_memory_TMA: tabla con los datos de memoria total asignada (TotalMemAllocated) por método.
  - T_memory_TMF: tabla con los datos de memoria total liberada (TotalMemFreed) por método.
  - T_memory_PM: tabla con los datos de memoria pico (PeakMem) por método.
- aam_LinCoLiNe_used: nombre del fichero AAM utilizado en el método LinCoLiNe.
- aam_CoLiNe_used: nombre del fichero AAM utilizado en el método CoLiNe.

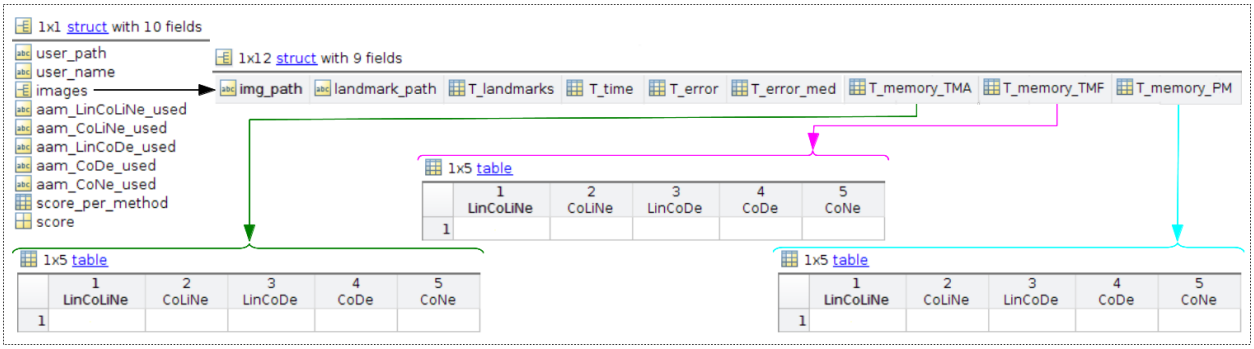


Figura 4.9: Organización interna ImagesDATA fase testeo

- aam_LinCoDe_used: nombre del fichero AAM utilizado en el método LinCoDe.
- aam_CoDe_used: nombre del fichero AAM utilizado en el método CoDe.
- aam_CoNe_used: nombre del fichero AAM utilizado en el método CoNe.

Los distintos ImagesDATA generados, 1 por cada uno de los 23 usuarios, se nombran, dependiendo si estamos tomando mediciones de memoria utilizada o no, siguiendo el patrón:

ImagesDATA_+nombre de usuario_+aam_optimized
ImagesDATA_+nombre de usuario_+aam_optimized_memory

mat profileDATA
Esta estructura es generada al activar la función profile de MATLAB®. En ella se guarda información sobre los programas ejecutados, funciones utilizadas, memoria... Posteriormente los extraemos y añadimos los más relevantes a una estructura ImagesDATA.

Capítulo 5

Análisis de los resultados

5.1. Fase de optimización

Debido a la gran cantidad de datos obtenidos en la fase de optimización vamos a agruparlos por parámetros, en una serie de diagramas de caja (figura 5.1) para intentar ver si hay alguno de ellos que resulte decisivo en la creación de AAM óptimos.

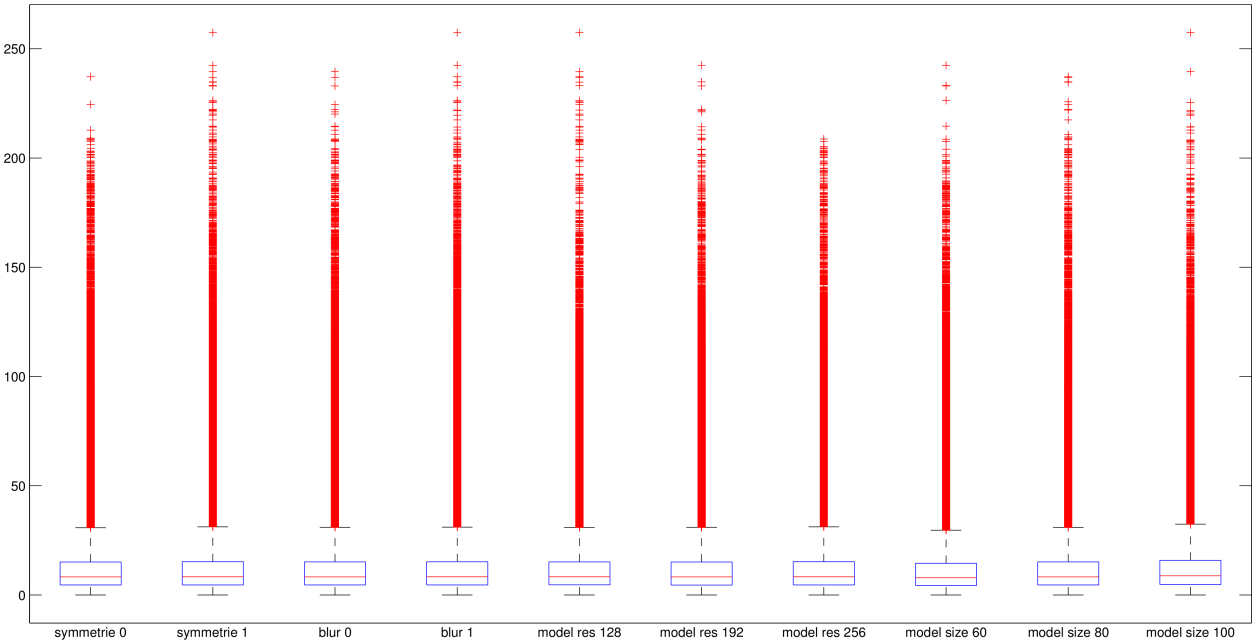


Figura 5.1: Boxplot distancia euclídea (píxeles).Fase optimización, por parámetros

En estos diagramas de caja podemos observar la mediana de los valores (línea roja horizontal) y los percentiles 25 y 75, que conforman la caja (en azul). Los "bigotes" que parten de la caja indican los valores mínimos y máximos (líneas negras horizontales). Los valores atípicos obtenidos están marcados con cruces rojas.
A la vista de los resultados, quitando el parámetro del tamaño del modelo, que si parece que favorece claramente al valor 60, el resto no parece que sean concluyentes.
Vamos a representar en diagramas de caja los resultados agrupados por AAM y método. Los trasladaremos a unas tablas (marcados en verde están los valores mínimos en su categoría) para facilitar su visualización:

1. Método LinCoLiNe: figura 5.2, tabla 5.1

2. Método CoLiNe: figura 5.3, tabla 5.2

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

3. Método LinCoDe: figura 5.4, tabla 5.3
4. Método CoDe: figura 5.5, tabla 5.4
5. Método CoNe: figura 5.6, tabla 5.5

En esta representación si que atisbamos que hay AAM que se comportan de mejor manera que otros. Hemos marcado con un superíndice (del 1 al 4) los que consideramos mejores candidatos a utilizarse en la fase de testeo.

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat	0,07	4,55	8,21	14,43	29,24
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat	0,02	4,58	8,25	14,57	29,54
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	0,08	4,89	8,69	15,30	30,90
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	0,03	4,89	8,72	15,37	31,07
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	0,04	5,14	9,29	16,92	34,57
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	0,05	5,17	9,34	16,93	34,57
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat	0,02	4,80	8,47	15,68	31,96
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat	0,10	4,80	8,47	15,63	31,85
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	0,10	5,09	9,05	16,44	33,45
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	0,08	5,11	9,11	16,65	33,96
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	0,02	5,38	9,67	17,96	36,80
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	0,13	5,39	9,73	18,02	36,91
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[4]	0,10	4,47	8,09	14,55	29,67
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[2]	0,04	4,51	8,06	14,52	29,53
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	0,07	4,73	8,55	15,06	30,51
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	0,03	4,76	8,57	15,07	30,53
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	0,09	5,13	9,08	16,28	32,98
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	0,10	5,14	9,06	16,23	32,84
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	0,05	4,69	8,35	15,22	30,98
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	0,02	4,71	8,42	15,30	31,15
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	0,06	4,88	8,68	15,73	32,01
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	0,06	4,91	8,69	15,76	32,04
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	0,06	5,23	9,29	16,64	33,75
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	0,09	5,30	9,44	17,05	34,66
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat[1]	0,07	4,36	7,95	14,16	28,85
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat[3]	0,10	4,43	8,07	14,23	28,91
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	0,01	4,70	8,51	15,06	30,60
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	0,02	4,70	8,51	15,19	30,91
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	0,02	4,96	8,79	15,84	32,15
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	0,06	4,95	8,85	15,84	32,15
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat	0,10	4,62	8,34	15,02	30,60
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat	0,08	4,65	8,35	15,12	30,82
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	0,05	4,68	8,40	15,38	31,42
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	0,02	4,70	8,42	15,42	31,45
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	0,06	5,00	8,88	16,09	32,72
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	0,06	5,04	8,90	16,17	32,83

Tabla 5.1: Distancia euclídea (píxeles). Fase optimización, método LinCoLiNe

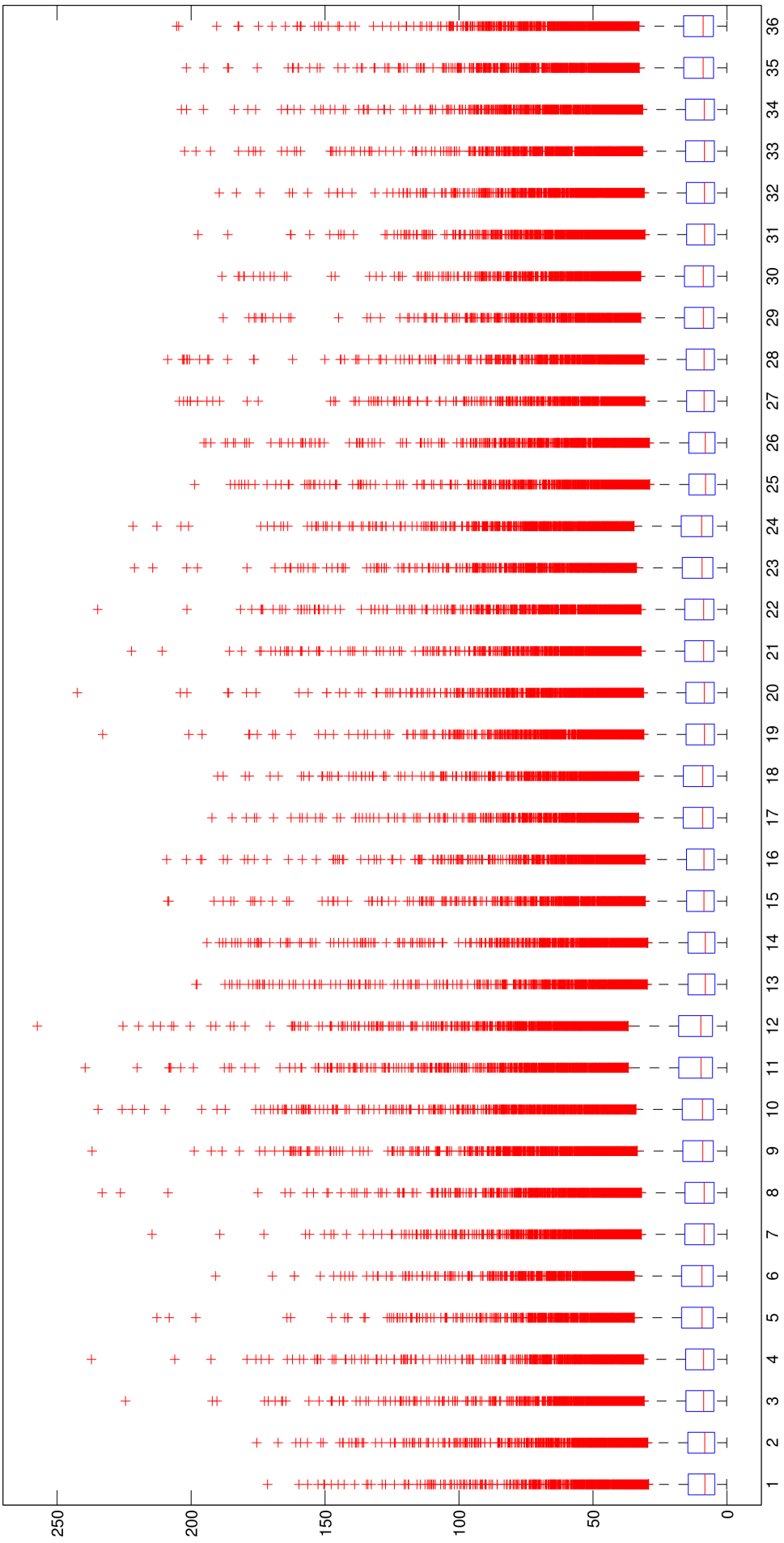


Figura 5.2: Boxplot distancia euclídea (píxeles).Fase optimización, método LinCoLiNe

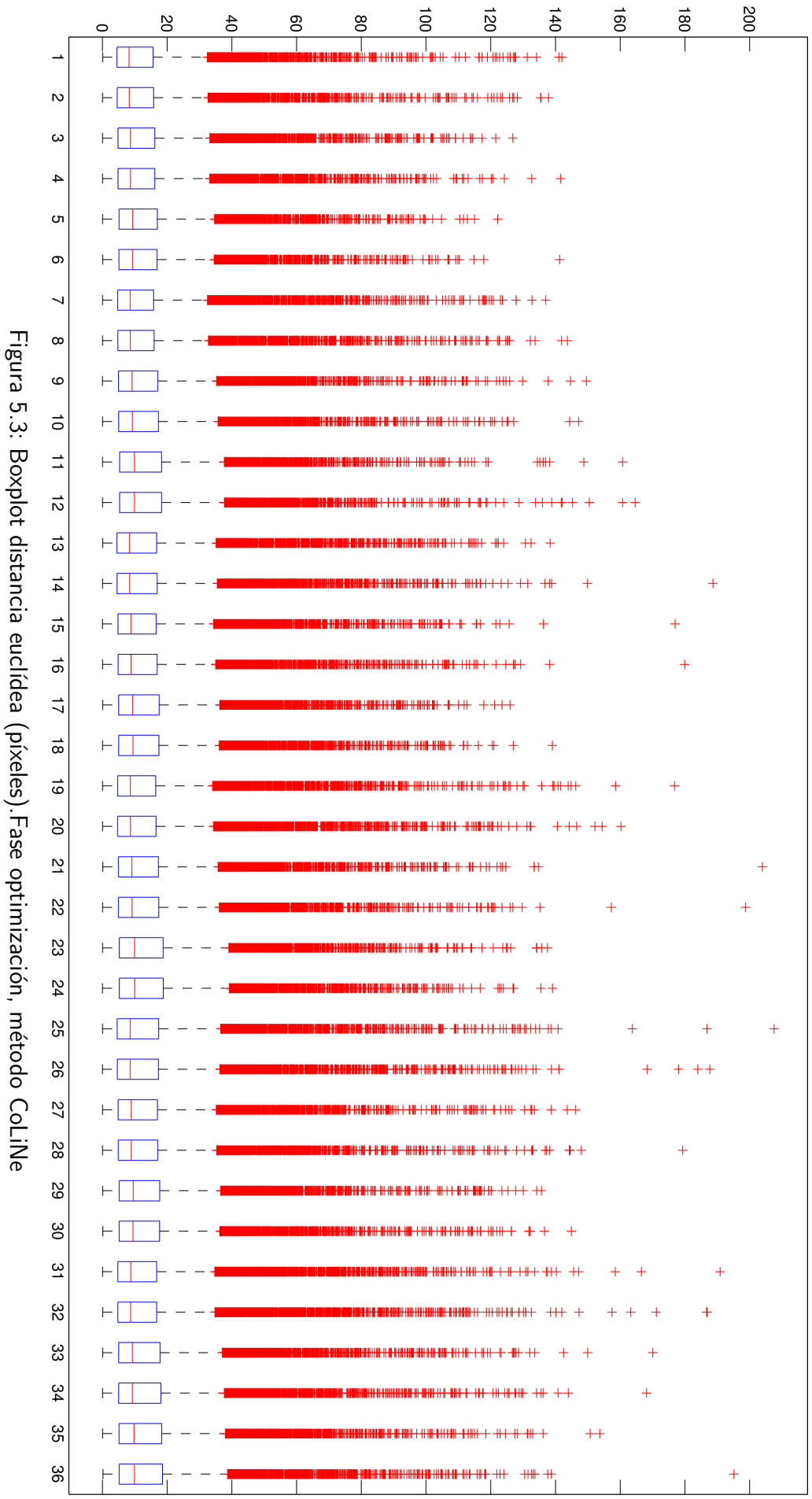


Figura 5.3: Boxplot distancia euclídea (píxeles).Fase optimización, método CoLiNe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat[1]	0,11	4,49	8,22	15,67	32,42
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[2]	0,03	4,56	8,31	15,81	32,63
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	0,04	4,82	8,66	16,18	33,21
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	0,05	4,83	8,66	16,18	33,19
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	0,04	5,15	9,37	16,95	34,65
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	0,11	5,11	9,29	16,90	34,53
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat	0,05	4,73	8,57	15,84	32,49
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat	0,04	4,76	8,63	15,97	32,78
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	0,02	4,97	9,16	17,10	35,30
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	0,05	5,00	9,23	17,31	35,77
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	0,06	5,33	9,89	18,27	37,68
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	0,07	5,30	9,90	18,28	37,71
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,08	4,50	8,37	16,78	35,16
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[4]	0,06	4,53	8,41	16,93	35,51
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	0,02	4,83	8,82	16,66	34,39
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	0,04	4,85	8,86	16,93	35,05
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	0,09	5,01	9,37	17,54	36,32
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	0,05	5,03	9,44	17,48	36,13
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	0,04	4,73	8,62	16,46	34,04
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	0,04	4,71	8,66	16,57	34,36
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	0,11	4,94	9,14	17,28	35,78
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	0,05	4,93	9,18	17,42	36,15
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	0,05	5,22	9,95	18,78	39,11
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	0,03	5,21	9,99	18,83	39,23
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	0,05	4,58	8,59	17,38	36,55
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat	0,02	4,61	8,58	17,30	36,32
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	0,05	4,85	8,90	17,00	35,21
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	0,05	4,88	8,94	17,07	35,34
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	0,08	5,18	9,51	17,72	36,53
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	0,14	5,19	9,44	17,65	36,28
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat	0,05	4,76	8,76	16,76	34,75
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat	0,09	4,79	8,72	16,78	34,77
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	0,08	5,00	9,24	17,85	37,09
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	0,03	5,02	9,24	18,09	37,68
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	0,05	5,16	9,77	18,31	38,01
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	0,17	5,18	9,90	18,59	38,68

Tabla 5.2: Distancia euclídea (píxeles). Fase optimización, método CoLiNe

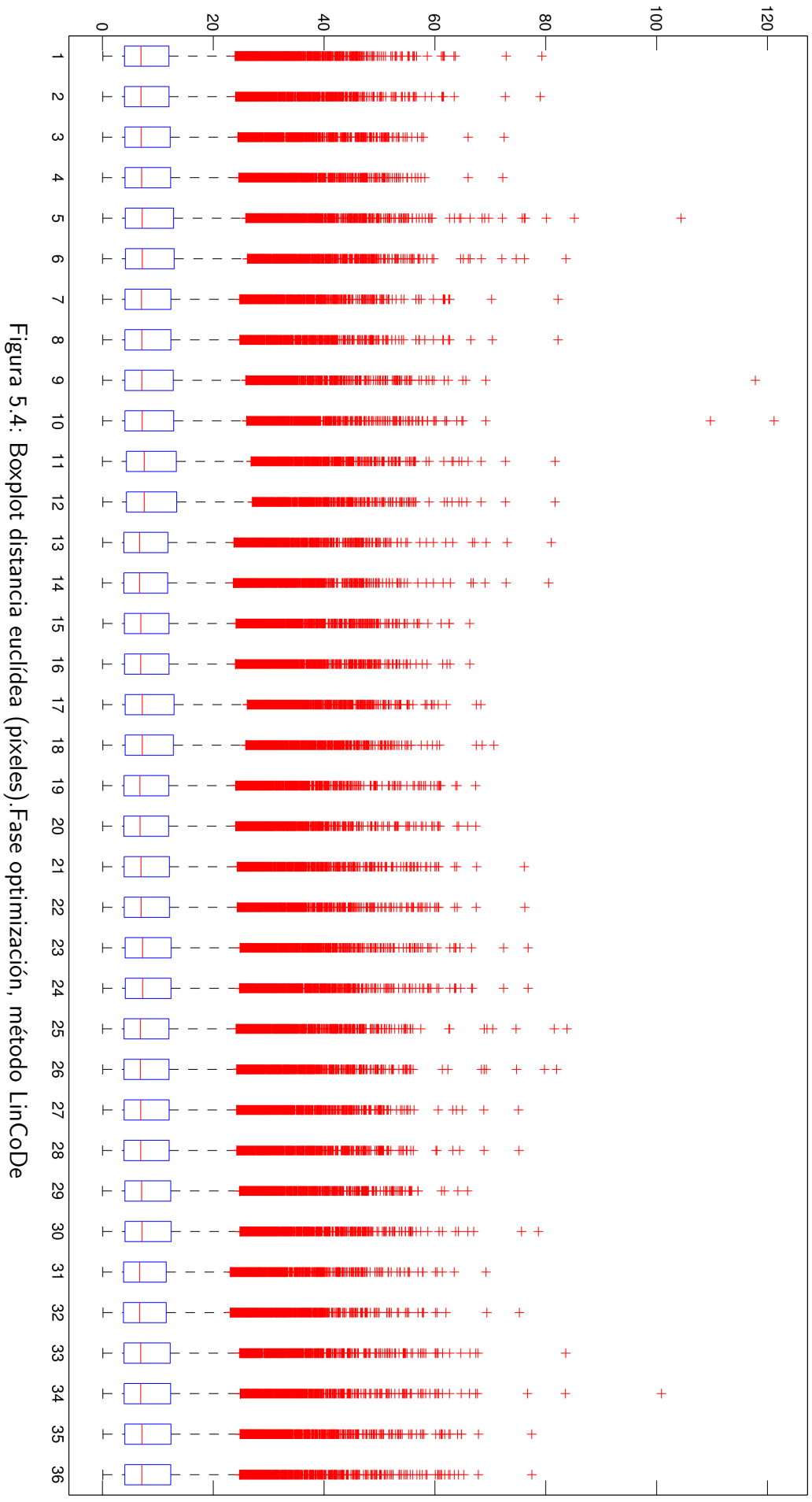


Figura 5.4: Boxplot distancia euclídea (píxeles).Fase optimización, método LinCoDe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat	0,08	3,97	6,96	11,98	23,95
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat	0,05	3,99	6,95	12,01	24,03
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	0,05	4,06	7,01	12,23	24,46
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	0,03	4,07	7,07	12,30	24,63
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	0,07	4,12	7,18	12,83	25,87
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	0,02	4,15	7,20	12,95	26,16
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat	0,05	4,09	7,05	12,36	24,77
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat	0,05	4,07	7,11	12,36	24,78
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	0,05	4,06	7,11	12,79	25,88
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	0,05	4,06	7,17	12,85	26,01
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	0,01	4,33	7,54	13,33	26,82
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	0,08	4,31	7,55	13,41	27,05
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,05	3,86	6,69	11,81	23,73
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[4]	0,01	3,86	6,70	11,77	23,64
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	0,09	3,97	6,94	12,02	24,09
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	0,03	3,99	6,91	12,01	24,00
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	0,08	4,09	7,20	12,92	26,15
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	0,08	4,11	7,19	12,82	25,87
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	0,08	3,88	6,78	11,96	24,06
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	0,06	3,87	6,79	11,95	24,05
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	0,06	3,92	6,96	12,09	24,33
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	0,07	3,93	6,97	12,11	24,35
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	0,04	4,10	7,23	12,39	24,83
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	0,07	4,12	7,26	12,37	24,74
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	0,04	3,89	6,84	11,98	24,12
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat	0,05	3,88	6,83	12,02	24,22
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	0,09	3,91	6,91	12,04	24,23
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	0,03	3,90	6,91	12,05	24,25
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	0,01	4,05	7,08	12,32	24,74
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	0,06	4,08	7,13	12,39	24,82
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat[2]	0,05	3,81	6,72	11,52	23,08
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat[1]	0,09	3,80	6,72	11,52	23,07
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	0,07	3,89	6,90	12,24	24,77
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	0,08	3,91	6,92	12,30	24,87
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	0,10	4,04	7,13	12,37	24,85
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	0,09	4,04	7,11	12,32	24,75

Tabla 5.3: Distancia euclídea (píxeles). Fase optimización, método LinCoDe

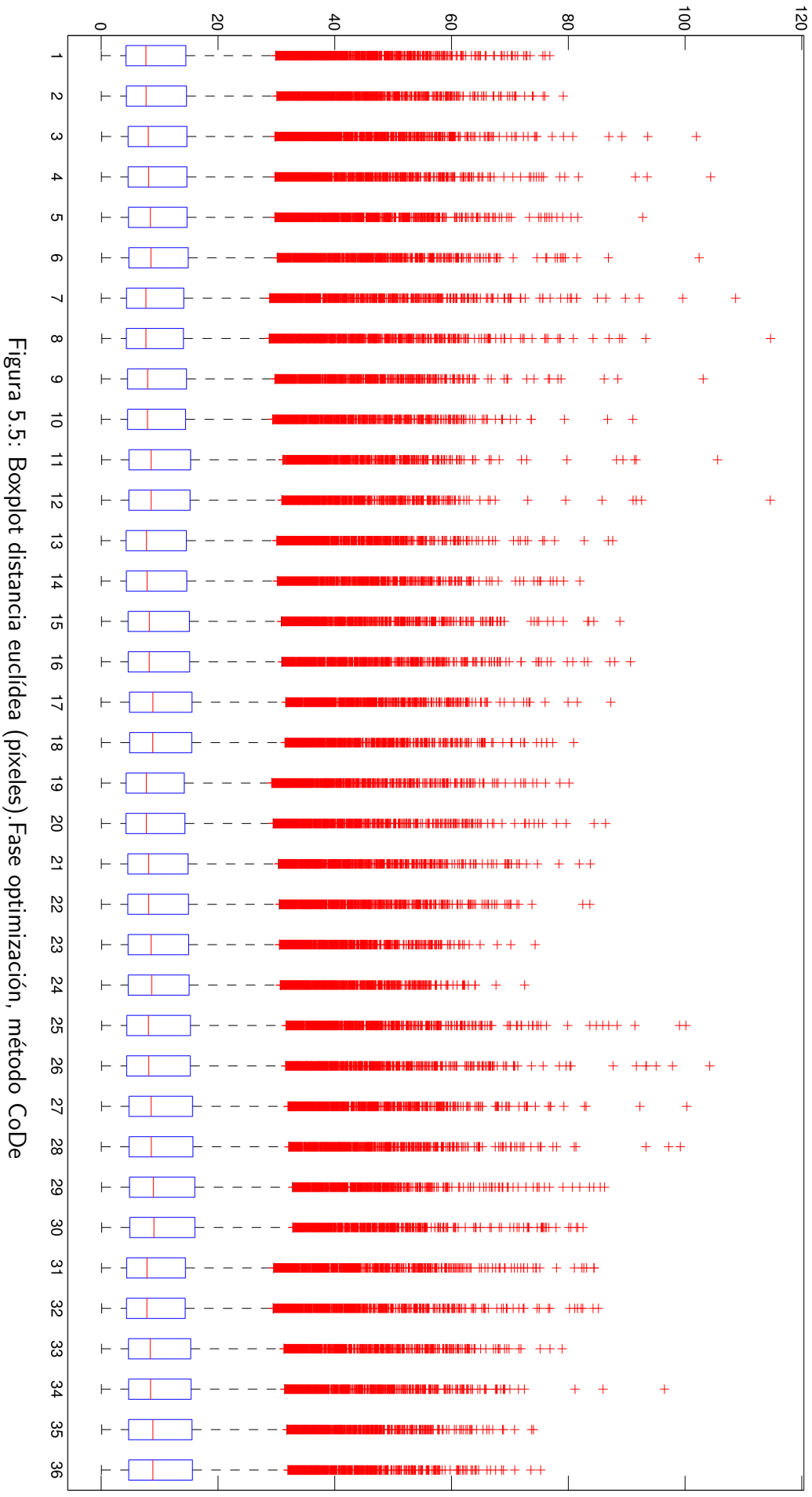


Figura 5.5: Boxplot distancia euclídea (píxeles).Fase optimización, método CoDe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat[3]	0,03	4,24	7,66	14,51	29,89
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[4]	0,01	4,31	7,68	14,62	30,09
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	0,09	4,62	8,07	14,68	29,77
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	0,04	4,60	8,12	14,67	29,78
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	0,07	4,69	8,42	14,71	29,75
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	0,10	4,74	8,52	14,92	30,15
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat[2]	0,03	4,33	7,66	14,15	28,85
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat[1]	0,12	4,31	7,67	14,09	28,77
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	0,03	4,51	7,96	14,60	29,74
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	0,03	4,52	7,93	14,46	29,37
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	0,08	4,76	8,58	15,30	31,11
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	0,06	4,75	8,55	15,23	30,93
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat	0,01	4,27	7,76	14,60	30,07
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat	0,03	4,31	7,86	14,66	30,18
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	0,06	4,62	8,26	15,11	30,84
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	0,05	4,62	8,23	15,16	30,93
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	0,00	4,85	8,85	15,57	31,62
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	0,05	4,88	8,82	15,52	31,48
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	0,05	4,26	7,73	14,24	29,20
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	0,04	4,24	7,74	14,33	29,46
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	0,01	4,56	8,10	14,88	30,37
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	0,03	4,56	8,12	14,95	30,52
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	0,08	4,61	8,56	14,98	30,53
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	0,06	4,64	8,66	15,05	30,67
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	0,09	4,37	8,10	15,29	31,66
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat	0,05	4,37	8,14	15,26	31,59
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	0,07	4,77	8,57	15,66	31,99
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	0,02	4,78	8,59	15,72	32,10
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	0,07	4,86	8,95	16,04	32,77
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	0,09	4,90	9,04	16,06	32,80
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat	0,06	4,35	7,86	14,42	29,51
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat	0,09	4,34	7,83	14,39	29,45
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	0,07	4,67	8,40	15,33	31,31
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	0,03	4,71	8,46	15,41	31,42
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	0,08	4,69	8,86	15,55	31,80
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	0,05	4,73	8,86	15,63	31,99

Tabla 5.4: Distancia euclídea (píxeles). Fase optimización, método CoDe

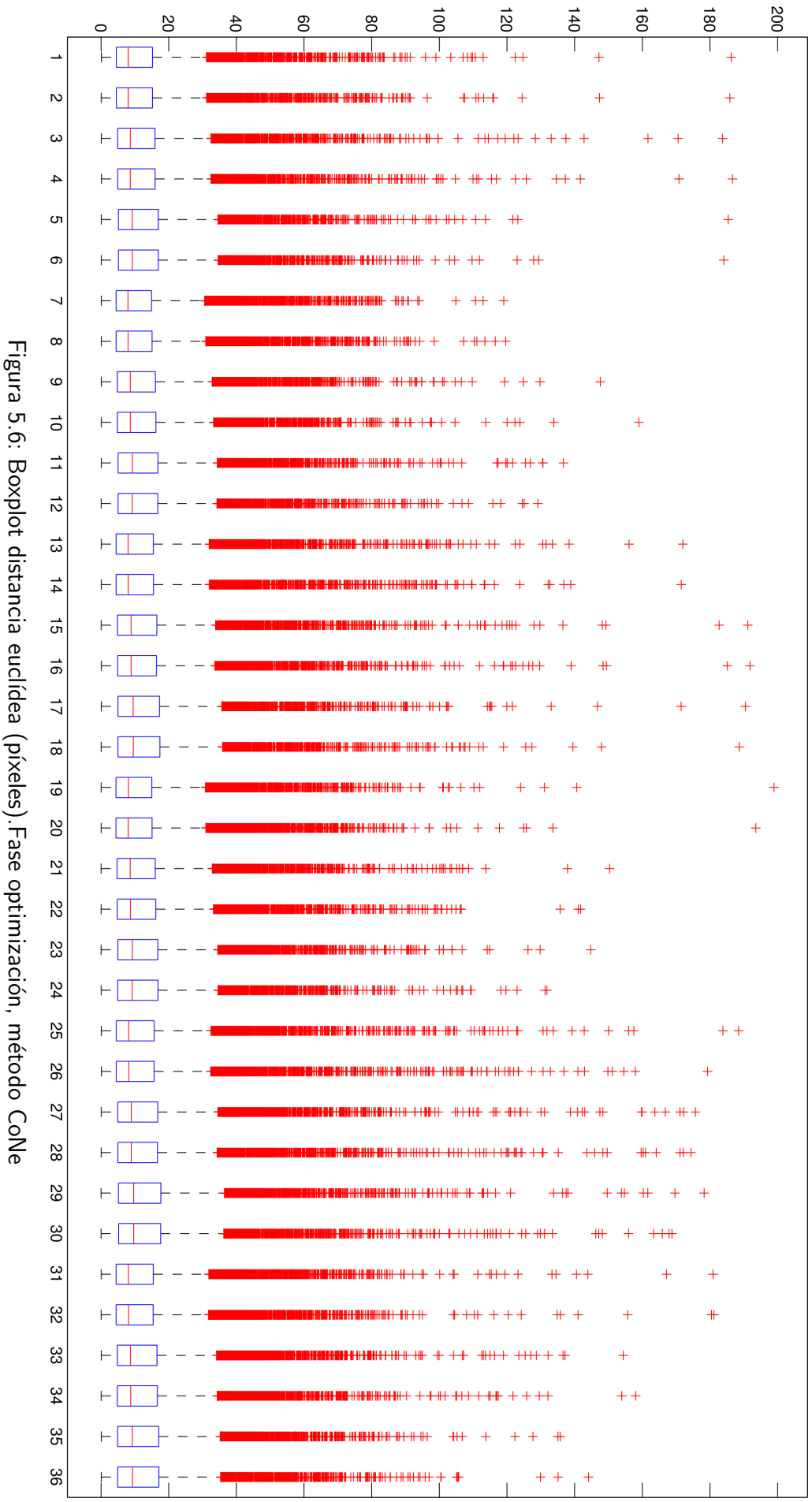


Figura 5.6: Boxplot distancia euclídea (píxeles).Fase optimización, método CoNe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat	0,12	4,53	8,05	15,19	31,18
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[4]	0,06	4,53	8,03	15,21	31,23
3	aam_color_mixed_aam_all_128_480_80_0_(60x8).mat	0,10	4,90	8,65	15,94	32,51
4	aam_color_mixed_aam_all_128_480_80_1_(60x8).mat	0,11	4,93	8,68	15,97	32,51
5	aam_color_mixed_aam_all_128_480_100_0_(60x8).mat	0,07	5,13	9,20	16,91	34,57
6	aam_color_mixed_aam_all_128_480_100_1_(60x8).mat	0,08	5,13	9,27	16,94	34,65
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat[1]	0,09	4,46	7,97	14,94	30,66
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat[2]	0,09	4,46	7,99	15,06	30,96
9	aam_color_mixed_aam_all_128_960_80_0_(60x8).mat	0,07	4,82	8,69	16,06	32,88
10	aam_color_mixed_aam_all_128_960_80_1_(60x8).mat	0,07	4,81	8,65	16,21	33,29
11	aam_color_mixed_aam_all_128_960_100_0_(60x8).mat	0,09	5,05	9,24	16,79	34,39
12	aam_color_mixed_aam_all_128_960_100_1_(60x8).mat	0,06	5,04	9,22	16,76	34,29
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,07	4,46	8,02	15,49	32,01
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat	0,07	4,47	8,03	15,50	32,04
15	aam_color_mixed_aam_all_192_480_80_0_(60x8).mat	0,03	4,84	8,91	16,49	33,88
16	aam_color_mixed_aam_all_192_480_80_1_(60x8).mat	0,08	4,90	8,92	16,40	33,64
17	aam_color_mixed_aam_all_192_480_100_0_(60x8).mat	0,08	5,01	9,48	17,30	35,72
18	aam_color_mixed_aam_all_192_480_100_1_(60x8).mat	0,03	5,01	9,52	17,39	35,94
19	aam_color_mixed_aam_all_192_960_60_0_(60x8).mat	0,04	4,41	8,06	15,02	30,91
20	aam_color_mixed_aam_all_192_960_60_1_(60x8).mat	0,06	4,43	8,05	15,07	31,03
21	aam_color_mixed_aam_all_192_960_80_0_(60x8).mat	0,02	4,75	8,61	16,01	32,90
22	aam_color_mixed_aam_all_192_960_80_1_(60x8).mat	0,06	4,76	8,69	16,17	33,27
23	aam_color_mixed_aam_all_192_960_100_0_(60x8).mat	0,02	4,95	9,24	16,79	34,54
24	aam_color_mixed_aam_all_192_960_100_1_(60x8).mat	0,04	4,93	9,21	16,80	34,61
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	0,08	4,51	8,17	15,71	32,51
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat	0,11	4,51	8,18	15,69	32,44
27	aam_color_mixed_aam_all_256_480_80_0_(60x8).mat	0,06	4,92	8,99	16,78	34,55
28	aam_color_mixed_aam_all_256_480_80_1_(60x8).mat	0,04	4,95	8,96	16,70	34,32
29	aam_color_mixed_aam_all_256_480_100_0_(60x8).mat	0,03	5,14	9,66	17,68	36,48
30	aam_color_mixed_aam_all_256_480_100_1_(60x8).mat	0,03	5,19	9,65	17,63	36,31
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat	0,04	4,45	8,08	15,43	31,90
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat	0,04	4,45	8,10	15,40	31,81
33	aam_color_mixed_aam_all_256_960_80_0_(60x8).mat	0,07	4,79	8,70	16,55	34,18
34	aam_color_mixed_aam_all_256_960_80_1_(60x8).mat	0,04	4,80	8,79	16,63	34,38
35	aam_color_mixed_aam_all_256_960_100_0_(60x8).mat	0,10	4,89	9,24	17,04	35,25
36	aam_color_mixed_aam_all_256_960_100_1_(60x8).mat	0,09	4,90	9,28	17,08	35,36

Tabla 5.5: Distancia euclídea (píxeles). Fase optimización, método CoNe

5.2. Fase de testeo

Lo primero que tenemos que realizar es decidir con que AAM vamos a trabajar en esta fase. En la anterior obteníamos 4 candidatos a AAM óptimo para cada método (marcados con un superíndice) y debemos decantarnos por uno.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

5.2.1. Elección mejores AAM por método

Después de observar que valores son los significativos: mediana, percentiles, valores mínimo y máximo, decidimos escoger aquellos AAM que presentasen una mediana más baja. Para la elección utilizamos cada AAM sobre los 23 usuarios restantes de la base de datos, obteniendo los resultados que se pueden ver en las tablas 5.6, 5.7, 5.8, 5.9 y 5.10

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat[1]	0,09	5,04	8,97	17,91	37,20
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[2]	0,07	5,18	9,32	18,14	37,56
26	aam_color_mixed_aam_all_256_480_60_1_(60x8).mat[3]	0,06	5,04	9,02	17,90	37,18
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[4]	0,08	5,17	9,25	18,06	37,40

Tabla 5.6: Distancia euclídea (píxeles). Fase testeo, método LinCoLiNe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat[1]	0,06	5,24	9,52	18,04	37,25
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[2]	0,05	5,20	9,56	17,90	36,95
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,01	5,15	9,38	17,93	37,03
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[4]	0,07	5,20	9,45	17,96	37,10

Tabla 5.7: Distancia euclídea (píxeles). Fase testeo, método CoLiNe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
32	aam_color_mixed_aam_all_256_960_60_1_(60x8).mat[1]	0,04	4,11	7,46	13,40	27,32
31	aam_color_mixed_aam_all_256_960_60_0_(60x8).mat[2]	0,09	4,09	7,49	13,43	27,44
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,05	4,12	7,30	12,93	26,15
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat[4]	0,04	4,11	7,29	12,93	26,17

Tabla 5.8: Distancia euclídea (píxeles). Fase testeo, método LinCoDe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat[1]	0,03	4,66	8,22	14,86	30,15
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat[2]	0,09	4,66	8,22	14,85	30,11
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat[3]	0,08	4,65	8,21	14,96	30,37
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[4]	0,03	4,64	8,25	14,97	30,46

Tabla 5.9: Distancia euclídea (píxeles). Fase testeo, método CoDe

	AAM (.mat file)	$V_{mín}$	P_{25}	P_{50}	P_{75}	$V_{máx}$
7	aam_color_mixed_aam_all_128_960_60_0_(60x8).mat[1]	0,12	4,89	8,75	16,12	32,91
8	aam_color_mixed_aam_all_128_960_60_1_(60x8).mat[2]	0,03	4,87	8,75	16,11	32,95
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat[3]	0,04	4,74	8,72	16,40	33,88
2	aam_color_mixed_aam_all_128_480_60_1_(60x8).mat[4]	0,08	4,83	8,86	16,40	33,75

Tabla 5.10: Distancia euclídea (píxeles). Fase testeo, método CoNe

En la tabla 5.11 podemos ver cuales son los AAM óptimos por método (marcados en verde en las tablas anteriores).
Se observa que el método LinCoDe es aquel que posee unos valores más bajos entre los AAM óptimos, por lo que lo consideraremos el mejor método. El método CoDe es el siguiente mejor (ver figura 5.7). Cabe destacar que los resultados obtenidos en esta fase son peores que los obtenidos en la de optimización, ya que hay varios usuarios que el programa no es capaz de ajustar correctamente.
Se observa que los AAM número 13 y 14 (sólo se diferencian en que uno tiene activo el desenfoque y el otro no) están presentes como posibles AAM óptimos en prácticamente todos los métodos, salvo en el método CoDe, siendo ganadores en 3 métodos distintos: CoLiNe, LinCoDe y CoNe. Si solamente se pudiera escoger un AAM para cualquier método se debería escoger el número 13, o en su defecto el 14. Recordemos sus parámetros de creación:

- Resolución del modelo: 192 píxeles.
- Tamaño del modelo: 60
- Blur: óptimo si está desactivado (aunque no influye de manera decisiva).
- Simetría: sin activar.

	AAM (.mat file)	Método
25	aam_color_mixed_aam_all_256_480_60_0_(60x8).mat	LinCoLiNe
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat	CoLiNe
14	aam_color_mixed_aam_all_192_480_60_1_(60x8).mat	LinCode
1	aam_color_mixed_aam_all_128_480_60_0_(60x8).mat	CoDe
13	aam_color_mixed_aam_all_192_480_60_0_(60x8).mat	CoNe

Tabla 5.11: Fase testeo, AAM óptimos

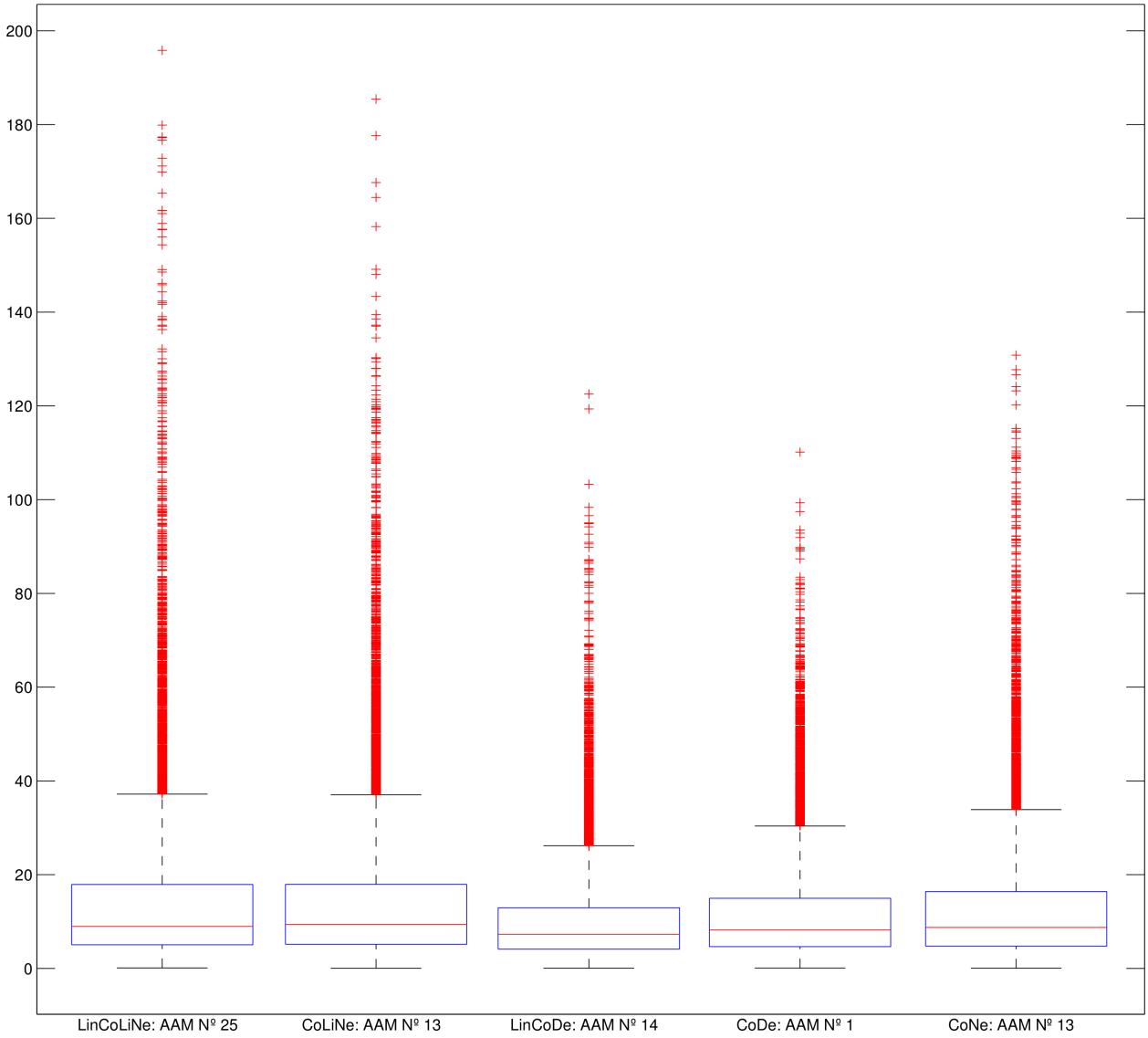


Figura 5.7: Boxplot AAM óptimos por método

5.2.2. Consumo de tiempo y memoria RAM

En la tabla 5.12 se puede ver el tiempo medio que tarda cada método en analizar una imagen con los AAM óptimos. Este tiempo medido es la media obtenida sobre 552 imágenes (23 usuarios, con 12 imágenes por usuario, y lo hemos hecho dos veces), por lo que lo podemos considerar como representativo.

El método LinCoDe, además de ser el más eficaz es el más rápido de los 5 estudiados, seguido por el método CoDe. Hay una pequeña diferencia a favor de la ejecución del programa en Windows 7 contra la ejecución en GNU/Linux, salvo en el método CoNe, en el que el tiempo es sensiblemente inferior en este último SO.

El intento de añadir partes de código que enlazaran con la biblioteca externa desarrollada en C++ ha sido una decepción. No hemos podido añadir más que un fichero MEX, ya que el resto provocaban que el programa MATLAB® diese un error y se cerrase. En GNU/Linux directamente no hemos podido añadir ninguno, porque incluso aquel que funcionaba bien bajo Windows provocaba el cierre del programa. Lo probamos con las versiones R2011a y R2014a de MATLAB® (con la versión R2014a directamente no funcionaba ninguno). Este es el motivo por el que no aparecen datos de la versión MEX en GNU/Linux. Además, como se puede comprobar en la tabla 5.12, añadir esa clase de ficheros provoca una penalización en el tiempo de ejecución.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

	LinCoLiNe	CoLiNe	LinCoDe	CoDe	CoNe
Windows 7	3,44	4,96	1,51	2,00	19,28
Windows 7 (MEX)	5,30	7,78	2,60	3,21	21,30
GNU/Linux	3,48	5,04	1,59	2,14	16,35

Tabla 5.12: Tiempo (s) por método

El uso de memoria lo podemos calificar como intensivo. Vamos a obviar el hecho que el programa MATLAB® necesita gran cantidad de memoria para operar (aproximadamente unos 480 MB en el SO GNU/Linux y unos 430 en el caso de Windows, sólo para tener operativo el programa). Hay que tener en cuenta que los AAM utilizados para cada uno de los métodos son diferentes (aunque en algún caso se repita), por lo que las necesidades de memoria de cada uno de ellos también lo son. En la tabla 5.13 podemos ver dichas necesidades. La memoria consumida depende del número de imágenes utilizadas en la creación del AAM, su resolución... Como anécdota citaremos que la carga del AAM, con mayor requerimiento de memoria de entre los 36 generados en la fase de de entrenamiento, necesita 802 MB en el SO GNU/Linux.

	LinCoLiNe	CoLiNe	LinCoDe	CoDe	CoNe
Windows 7	524	333	337	154	336
Windows 7 (MEX)	525	333	336	155	336
GNU/Linux	596	374	376	164	376

Tabla 5.13: Memoria utilizada (MB) para cargar AAM por método

Una vez tenemos el AAM ya cargado, el resto de los procesos necesarios para realizar los ajustes necesitan también una cantidad de memoria considerable, como podemos ver en la tabla 5.14. La memoria total utilizada por cada método sería la suma de los valores de ambas tablas, pero consideramos interesante separar la necesaria para cargar el AAM de la necesaria para su implementación. En cualquier caso en GNU/Linux se utiliza más memoria que en Windows. Por su parte el uso de los ficheros MEX no implica un aumento muy significativo en el uso de memoria (unos 4 MB aproximadamente).

	LinCoLiNe	CoLiNe	LinCoDe	CoDe	CoNe
Windows 7	70,1	115,0	68,1	63,7	114,5
Windows 7 (MEX)	74,3	119,4	71,9	66,9	118,6
GNU/Linux	98,3	127,6	90,1	80,8	130,4

Tabla 5.14: Memoria utilizada (MB) por método

El método que necesita menos memoria es el CoDe, en ambas vertientes: el AAM óptimo utilizado para dicho método es el más liviano de todos y su algoritmo necesita menos memoria para su uso. El siguiente método a escoger por este parámetro es LinCoDe que también tiene un consumo de memoria contenido en su implementación, pero su AAM óptimo si que presenta un consumo de RAM muy importante.

Capítulo 6

Conclusiones

La migración de plataforma del software no es muy complicada, ya que la mayor parte del mismo ya es multiplaforma, al estar escrita en lenguajes, MATLAB® y C++, presentes en ambos Sistemas Operativos. La utilización de librerías que, si bien no se incluyen en el estándar, son tan utilizadas que están portadas a las principales plataformas, también ayuda.
Los cambios que hay que introducir en el código escrito en C++ para que funcione en Windows no son muy profundos, sobre todo desde que los compiladores de Microsoft son compatibles con la especificación C++11. La introducción de dicho estándar en el programa Microsoft Visual Studio 2013 Professional ha facilitado el trabajo. Eso si, el código escrito en C++ es más rápido ejecutándose sobre GNU/Linux que sobre Windows.
El código escrito en MATLAB® directamente funciona tal cual en cualquier plataforma soportada por el programa, salvo que se utilice alguna función exclusiva de un Sistema Operativo concreto, lo que no es el caso salvo el apartado de exportación de los datos a excel, que sólo funciona en Windows.
Prácticamente no hemos podido añadir código que haga uso de la librería en C++ desde el programa MATLAB®, ya que provocaba cierres inesperados del mismo. Además su inclusión provoca una penalización en la velocidad de ejecución. Esta ha sido la parte más frustrante del proyecto.
Para poder adaptar el software a la base de datos GI4E ha sido necesario realizar varias modificaciones a la misma:

- Creación de imágenes de menor tamaño (desechando el fondo de las mismas, quedándonos sólo con la cara).
- Creación de máscaras de imagen a partir de los landmarks marcados de manera manual.
- Creación de documentos de landmarks separados, con un formato concreto.

Todos los ficheros generados tienen que guardarse junto al resto de los ficheros de la base de datos para el correcto funcionamiento de las fases de entrenamiento, optimización y testeo.
Para cada uno de los métodos implementados, puede haber un AAM óptimo distinto. En este caso, en dos de ellos coincide el mismo (ver tabla 5.11). Si solamente se pudiera escoger un AAM para utilizarlo con cualquiera de los métodos implementados, se debería escoger el número 13 de los 36 creados o, en su defecto, el número 14. Esto puede ser útil cuando se utilice la versión gráfica del programa de testeo, ya que sufre esta limitación.
Los parámetros de creación del AAM número 13 son los siguientes:

- Resolución del modelo: 192 píxeles.
- Tamaño del modelo: 60

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- Blur: óptimo si está desactivado (aunque no influye de manera decisiva).
- Simetría: sin activar.

	LinCoLiNe	**CoLiNe**	**LinCoDe**	**CoDe**	**CoNe**
Eficiencia	4º	5º	1º	2º	3º
Velocidad	3º	4º	1º	2º	5º
Uso memoria	5º	3º	2º	1º	4º

Tabla 6.1: Clasificación mejores métodos por parámetros

En la tabla 6.1 podemos ver los distintos algoritmos clasificados según los distintos parámetros analizados. Si se tuviese que utilizar un sólo método deberemos escoger LinCoDe, salvo que el uso de memoria sea crítico, en cuyo caso deberemos decantarnos por CoDe, aunque los resultados son inferiores. Los nuevos métodos de ajuste presentados por los autores con la publicación de este software son claramente más eficientes que los que había en su momento.
Los resultados obtenidos son idénticos en ambos Sistemas Operativos. No se detectan grandes diferencias, ni de de velocidad ni de consumo de memoria, dependiendo de en cual de ellos se ejecute el programa. En Windows, en general, se ejecuta más rápido y con menores requerimientos de memoria que en GNU/Linux. Suponemos que el programa MATLAB® no está tan optimizado en dicho Sistema Operativo.
Esta implementación de AAM tiene unos requerimientos, respecto a requisitos computacionales y de necesidad de memoria, elevados. El uso de memoria en la fase de entrenamiento es especialmente alta, ya que ha llegado a utilizar hasta 6 GB (toda la que había disponible más 3,5 GB de intercambio) para la creación de alguno de los AAM con resoluciones más altas. Los requisitos computacionales son críticos en las fases de optimización y testeo (sobre todo en optimización), por lo que la posibilidad de utilizar equipos más potentes siempre será positiva.

Capítulo 7

Bibliografía

On Compositional Image Alignment with an Application to Active Appearance Models
Brian Amberg, Andrew Blake and Thomas Vetter
IN: CVPR'09, IEEE International Conference on Computer Vision and Pattern Recognition, Miama, USA, June. 2009

Comparing Active Shape Models with Active Appearance Models
T.F. Cootes, G. Edwards and C.J. Taylor
Dept. Medical Biophysics, Manchester University, UK

Active Appearance Models Revisited
Iain Matthews and Simon Baker
The Robotics Institute Carnegie Mellon University

Páginas web consultadas
http://es.mathworks.com/matlabcentral/
http://www.mathworks.es/support/
http://stackoverflow.com/
http://www.rad.upenn.edu/sbia/software/gondola/install/gcc.html
http://sourceforge.net/projects/boost/
http://icl.cs.utk.edu/lapack-for-windows/lapack/
http://www.fftw.org/install/windows.html
http://www.fdi.ucm.es/profesor/luis/fp/devtools/devtools.html

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Apéndice A

Software utilizado

Vamos a especificar como instalar el software necesario para la realización del proyecto.

A.1. GNU/Linux - Ubuntu 14.04 LTS 64 bits

A.1.1. Instalación compiladores y librerías necesarias

Para la instalación, tanto de los programas como de las librerías requeridas, utilizaremos las herramientas integradas en el SO, en este caso concreto, el gestor de paquetes apt-get.

- Abre un terminal.
- Para la instalación de todo lo necesario para la compilación y uso del programa ejecuta:

```
$ sudo apt-get install doxygen doxygen-latex lacheck latex-beamer
  latex-xcolor lmodern luatex pgf preview-latex-style prosper ps2eps
  tex-common texlive-base texlive-binaries texlive-base
  texlive-extra-utils texlive-font-utils texlive-generic-recommended
  texlive-latex-base texlive-latex-base-doc texlive-latex-extra
  texlive-latex-extra-doc texlive-latex-recommended
  texlive-latex-recommended-doc texlive-luatex texlive-pictures
  texlive-pictures-doc texlive-pstricks texlive-pstricks-doc
  libboost-filesystem-dev libboost-filesystem1.54-dev
  libboost-filesystem1.54.0 libboost-program-options1.54-dev
  libboost-program-options1.54.0 libboost-system-dev
  libboost-system1.54-dev libboost-system1.54.0 libblas-dev libblas3gf
  liblapack-dev liblapack3gf cmake gcc g++ g++-4.6 libstdc++6-4.6-dev
  libboost-dev libgfortran3 gfortran gfortran-4.6 libfftw3-dev
```

- Para la instalación de lo necesario para ejecutar la librería y los programas, pero sin la necesidad de compilarlos, ejecuta:

```
$ sudo apt-get install libboost-filesystem1.54.0
  libboost-program-options1.54.0 libboost-system1.54.0 libblas3gf
  liblapack3gf
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

A.1.2. Instalación MATLAB® R2011a y R2014a

En GNU/Linux tenemos que realizar 2 instalaciones distintas de MATLAB®:

- MATLAB® R2011a: versión con la que se desarrollo el software de manera original.
- MATLAB® R2014a: versión en la que desarrollamos la adaptación del software. Esto es debido a los requerimientos provocados por la utilización del programa Microsoft Visual Studio 2013 Professional para compilar en el entorno Windows.

Para la instalación del programa MATLAB® R2011a seguiremos las instrucciones que proporciona el propio instalador, especificando que la carpeta de instalación debe ser /usr/local/MATLAB/R2011a/.
Una vez terminada la instalación hay que modificar el fichero ~/.profile siguiendo las siguientes instrucciones:

1. Abre una terminal.
2. Edita el fichero ~/.profile:

   ```
   $ nano ~/.profile
   ```

3. Añade al final del fichero los siguientes comandos:

   ```
   export MATLABROOT=/usr/local/MATLAB/R2011a/
   PATH=$PATH:/usr/local/MATLAB/R2011a/bin
   ```

4. Salva las cambios pulsando las teclas CTRL+X.

Reinicia el equipo para que los cambios se hagan efectivos.
NOTA: para instalar la versión R2014a se puede seguir las instrucciones literalmente cambiando R2011a por R2014a.

A.2. Windows 7 Professional 64 bits

A.2.1. Instalación Microsoft Visual Studio 2013 Professional

Se puede descargar de la página:

`https://msdn.microsoft.com/es-es/library/dd831853.aspx`

Este programa es de pago, pero esta disponible de manera gratuita para estudiantes mediante registro en la web.
Hacemos una instalación personalizada, desmarcando todas las opciones salvo:

✓ "Microsoft Foundation Classes para C++"

Para poder ejecutar los programas compilados mediante Microsoft Visual Studio 2013 Professional hay que tener instalado en el ordenador "Paquetes redistribuibles de Visual C++ para Visual Studio 2013" disponibles en la página web para su descarga:

`http://www.microsoft.com/es-es/download/details.aspx?id=40784`

En concreto, para los programas creados en este proyecto, necesitamos la versión de 64 bits (vcredist_x64.exe). Cuando instalas el programa Microsoft Visual Studio 2013 Professional se instalan de manera automática.

A.2.2. Instalación MATLAB® R2014a

Para la instalación del programa MATLAB® R2014a seguiremos las instrucciones que proporciona el propio instalador.

A.2.3. Obtención librerías necesarias

En los siguientes apartados vamos a explicar como conseguir versiones operativas de las librerías requeridas. Unas veces hay que compilarlas desde el propio código fuente, en otros casos se pueden descargar ya compiladas y en otras veces hay que hacer pequeñas adaptaciones.

A.2.3.1. Librerías BOOST

Descarga

Utilizamos la última versión estable disponible (código fuente) en la dirección:

`http://sourceforge.net/projects/boost/files/boost/1.57.0/`

Compilación

1. Descomprime el contenido del fichero zip descargado en la carpeta c:\boost\boost_1_57_0
2. Abre una terminal con los compiladores de MS (Símbolo del sistema de las herramientas nativas de VS2013 x64)
3. Accede a la carpeta donde hemos descomprimido las librerías.

   ```
   > cd c:\boost\boost_1_57_0
   ```

4. Ejecuta la orden para configurar la compilación:

   ```
   > bootstrap.bat
   ```

5. Ejecuta la orden para compilar las librerías necesarias para 64 bits, versiones debug y release, versiones estáticas y dinámicas, multithreading:

   ```
   > b2.exe --prefix=c:\boost\boost64_157_msvc_2013 toolset=msvc
     address-model=64 --with-filesystem --with-system
     --with-program_options variant=debug,release
     link=static,shared threading=multi install
   ```

Una vez terminada la compilación, en la carpeta c:\boost\boost64_157_msvc_2013 habrá las siguientes librerías estáticas para su uso con el programa MSVS 2013:

- libboost_filesystem-vc120-mt-gd-1_57.lib: librería BOOST filesystem versión debug
- libboost_program_options-vc120-mt-gd-1_57.lib: librería BOOST program_options versión debug
- libboost_system-vc120-mt-gd-1_57.lib: librería BOOST system versión debug

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- libboost_filesystem-vc120-mt-1_57.lib: librería BOOST filesystem versión release
- libboost_program_options-vc120-mt-1_57.lib: librería BOOST program_options versión release
- libboost_system-vc120-mt-1_57.lib: librería BOOST system versión release

A.2.3.2. Librerías BLAS-LAPACK

Descarga

Utilizamos una versión de las librerías (compiladas para Windows 64 bits) en la dirección:

`http://icl.cs.utk.edu/lapack-for-windows/lapack/`

Contiene las siguientes librerías estáticas que funcionan en el programa MSVS 2013:

- libblas.lib
- liblapack.lib

Como estas librerías están compiladas con el programa MinGW, tienes que añadir al path del sistema (o dejar una copia en la carpeta del ejecutable) las librerías GNU siguientes:

- libgfortran-3.dll
- debuglibgcc_s_dw2-1.dll
- libgcc_s_seh-1.dll
- libgcc_s_sjlj-1.dll
- libquadmath-0.dll
- libwinpthread-1.dll

Además de las librerías BLAS y LAPACK propiamente dichas:

- libblas.dll
- liblapack.dll

A.2.3.3. Librerías fftw3

Descarga

Utilizamos una versión disponible (precompilada para Windows 64 bits) en la dirección:

`http://www.fftw.org/install/windows.html`

Compilación

Estas librerías están compiladas utilizando MinGW. Hay que recompilarlas para obtener las librerías .lib y poder enlazarlas con el compilador de Microsoft. Para ello:

1. Abre una terminal con los compiladores de MS (Símbolo del sistema de las herramientas nativas de VS2013 x64)
2. Utilizando el programa incluido en Visual Studio lib.exe, ejecuta:

```
> lib /machine:x64 /def:libfftw3-3.def
> lib /machine:x64 /def:libfftw3l-3.def
> lib /machine:x64 /def:libfftw3f-3.def
```

Se obtienen las librerías necesarias para su uso con el programa MSVS 2013:

- libfftw3-3.lib
- libfftw3f-3.lib
- libfftw3l-3.lib

Que se deben utilizar con sus correspondientes ficheros DLL.

- libfftw3-3.dll
- libfftw3f-3.dll
- libfftw3l-3.dll

Apéndice B

Especificaciones hardware equipo utilizado

El equipo utilizado para la realización del proyecto es de marca Acer, modelo Veriton M670G, situado en el "Laboratorio de proyectos, teoría de la señal y comunicación", con número de inventario 51442, que posee las siguientes características:

Procesador - Chipset		
	Marca	Intel®
	Modelo	Core™ 2 Quad Q8300
	Frecuencia uso	2500 MHz
	Caché L2	4MB
	Velocidad BUS sistema (FSB)	1333 MHz
	Intel® 64	Si
	Chipset	Intel® Q45 Express
Memoria RAM		
	Cantidad de memoria	4 GB
	Velocidad de reloj	1066 MHz
	Tipo	DDR3
	Slots de memoria	4 (2 libres)
Disco duro		
	Marca	Hitachi
	Modelo	HTS723232L9A360
	Capacidad	320GB
	Interfaz	SATA II
Tarjeta gráfica		
	Marca	nVidia®
	Modelo	Geforce® 9500 GT
	Frecuencia uso	550 MHz
	Cantidad de memoria	1 GB
	Tipo memoria	DDR3

Tabla B.1: Especificaciones técnicas equipo de pruebas

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

Apéndice C

Funciones de MATLAB® del software original

Vamos a enumerar la lista completa de archivos

- aam
  - aam_color_mixed_aam_all_192_912_60_1.code
  - aam_color_mixed_aam_all_192_912_60_1.code.lm_fitter
  - aam_color_mixed_aam_all_192_912_60_1.mat
- data
  - imm (en esta carpeta descomprimimos imm_face_db.tar.gz).
  - mixed_aam
    - 000_1_1.png.landmarks
    - 000_1_1.png
    - 000_1_1_mask.png
    - .
    - .
    - 328_4_2_mask.png
    - point_symmetrie
- data_other_licence
- examples
  - berlusconi.lms
  - berlusconi.ppm
  - example_click_landmarks.m
  - example_prepare_aam_for_libaam_code.m
- mex
  - aam_build_model.m
  - aam_build_model_mouth.m

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

- aam_covariance_svd.m
- aam_fit_code_mex.cpp
- aam_fit_to_warp_mex.cpp
- aam_gaussian_blur.m
- aam_gradient_fc_mex.cpp
- aam_load_code.cpp
- aam_prepare_code_fitter.m
- aam_prepare_compositional.m
- aam_prepare_warp_composition.m
- aam_procrustes.m
- aam_save_code.cpp
- aam_save_code_fitter.m
- aam_save_warp_fitter.cpp
- array.hpp
- ba_extrapolate_gridded.m
- ba_interp2.cpp
- ba_interp2.m

model

- aam_build_model.m
- aam_build_model_mouth.m
- aam_covariance_svd.m
- aam_display_warp.m
- aam_fc_gradient.m
- aam_fc_linear_gradient.m
- aam_fit_code.m
- aam_fit_coline.m
- aam_fit_cone.m
- aam_fit_gn.m
- aam_fit_lbfgs.m
- aam_fit_lincode.m

aam_fit_lincoline.m

aam_fit_lm.m

aam_fit_regularized.m

aam_fit_sd.m

aam_fit_to_pts.m

aam_fit_to_warp.m

aam_gaussian_blur.m

aam_ic_gradient.m

aam_load_image_and_parameters.m

aam_load_image_and_parameters_from_imm.m

aam_load_image_and_parameters_from_tf.m

aam_prepare_code_fitter.m

aam_prepare_compositional.m

aam_prepare_warp_composition.m

aam_procrustes.m

aam_render.m

aam_save_code_fitter.m

aam_to_html.m

aam_view_incremental_warps.m

ba_affine.cpp

ba_affine.m

ba_extrapolate_gridded.m

ba_interp2.cpp

ba_interp2.m

ba_rasterize.cpp

ba_rasterize.m

build_model.m

- build_models.m
- covariance_svd.m
- display_aam_with_params.m
- experiment_display_aam.m
- experiment_fit_imm.m
- gaussian_blur.m
- has_java.m
- rbf_calculate_distance_matrix.m
- rbf_create_basis.m
- rbf_evaluate.m
- rbf_find_coefficients.m
- vec.m

Apéndice D

Fichero log AAM generado en fase entrenamiento

```
aam file create with specifications:
Source data:
Database path: D:/BBDD
Training users list:
001, 003, 004, 008, 009, 010, 011, 013, 016, 018, 019, 020, 021, 022,
024, 025, 029, 030, 031, 036, 037, 040, 042, 044, 047, 049, 050, 051,
052, 053, 055, 056, 060, 061, 062, 067, 068, 070, 071, 072, 074, 075,
076, 078, 079, 081, 082, 083, 085, 088, 090, 091, 093, 095, 096, 097,
098, 100, 102, 103,
Test users list:
002, 006, 012, 015, 023, 027, 032, 033, 034, 035, 039, 041, 045, 046,
058, 059, 064, 065, 066, 080, 086, 087, 099,
Images per user: 8
Use crop images: 1
AAM specifications:
Model resolution: 192
Model size: 60
Use color: 1
Use blur: 0
Use symmetrie: 0
Loading data...
  0% D:/BBDD/001/images/001_02_crop.png
  0% D:/BBDD/001/images/001_05_crop.png
  1% D:/BBDD/001/images/001_06_crop.png
  1% D:/BBDD/001/images/001_07_crop.png
  1% D:/BBDD/001/images/001_09_crop.png
  1% D:/BBDD/001/images/001_10_crop.png
  1% D:/BBDD/001/images/001_11_crop.png
  2% D:/BBDD/001/images/001_12_crop.png
  2% D:/BBDD/003/images/003_02_crop.png
  2% D:/BBDD/003/images/003_03_crop.png
  2% D:/BBDD/003/images/003_05_crop.png
  3% D:/BBDD/003/images/003_06_crop.png
  3% D:/BBDD/003/images/003_07_crop.png
  3% D:/BBDD/003/images/003_08_crop.png
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

```
  3% D:/BBDD/003/images/003_09_crop.png
  3% D:/BBDD/003/images/003_11_crop.png
  4% D:/BBDD/004/images/004_03_crop.png
  4% D:/BBDD/004/images/004_05_crop.png
  4% D:/BBDD/004/images/004_06_crop.png
  4% D:/BBDD/004/images/004_07_crop.png
  4% D:/BBDD/004/images/004_09_crop.png
  5% D:/BBDD/004/images/004_10_crop.png
  5% D:/BBDD/004/images/004_11_crop.png
  5% D:/BBDD/004/images/004_12_crop.png
  5% D:/BBDD/008/images/008_01_crop.png
  5% D:/BBDD/008/images/008_02_crop.png
  6% D:/BBDD/008/images/008_04_crop.png
  6% D:/BBDD/008/images/008_06_crop.png
  6% D:/BBDD/008/images/008_08_crop.png
  6% D:/BBDD/008/images/008_10_crop.png
  6% D:/BBDD/008/images/008_11_crop.png
  7% D:/BBDD/008/images/008_12_crop.png
  7% D:/BBDD/009/images/009_01_crop.png
  7% D:/BBDD/009/images/009_02_crop.png
  7% D:/BBDD/009/images/009_04_crop.png
  8% D:/BBDD/009/images/009_06_crop.png
  8% D:/BBDD/009/images/009_07_crop.png
  8% D:/BBDD/009/images/009_08_crop.png
  8% D:/BBDD/009/images/009_10_crop.png
  8% D:/BBDD/009/images/009_12_crop.png
  9% D:/BBDD/010/images/010_01_crop.png
  9% D:/BBDD/010/images/010_02_crop.png
  9% D:/BBDD/010/images/010_04_crop.png
  9% D:/BBDD/010/images/010_06_crop.png
  9% D:/BBDD/010/images/010_08_crop.png
 10% D:/BBDD/010/images/010_09_crop.png
 10% D:/BBDD/010/images/010_10_crop.png
 10% D:/BBDD/010/images/010_12_crop.png
 10% D:/BBDD/011/images/011_02_crop.png
 10% D:/BBDD/011/images/011_03_crop.png
 11% D:/BBDD/011/images/011_04_crop.png
 11% D:/BBDD/011/images/011_05_crop.png
 11% D:/BBDD/011/images/011_07_crop.png
 11% D:/BBDD/011/images/011_08_crop.png
 11% D:/BBDD/011/images/011_09_crop.png
 12% D:/BBDD/011/images/011_10_crop.png
 12% D:/BBDD/013/images/013_01_crop.png
 12% D:/BBDD/013/images/013_04_crop.png
 12% D:/BBDD/013/images/013_05_crop.png
 13% D:/BBDD/013/images/013_07_crop.png
 13% D:/BBDD/013/images/013_08_crop.png
 13% D:/BBDD/013/images/013_10_crop.png
 13% D:/BBDD/013/images/013_11_crop.png
 13% D:/BBDD/013/images/013_12_crop.png
```

```
14% D:/BBDD/016/images/016_01_crop.png
14% D:/BBDD/016/images/016_02_crop.png
14% D:/BBDD/016/images/016_04_crop.png
14% D:/BBDD/016/images/016_05_crop.png
14% D:/BBDD/016/images/016_07_crop.png
15% D:/BBDD/016/images/016_08_crop.png
15% D:/BBDD/016/images/016_10_crop.png
15% D:/BBDD/016/images/016_11_crop.png
15% D:/BBDD/018/images/018_01_crop.png
15% D:/BBDD/018/images/018_03_crop.png
16% D:/BBDD/018/images/018_05_crop.png
16% D:/BBDD/018/images/018_07_crop.png
16% D:/BBDD/018/images/018_08_crop.png
16% D:/BBDD/018/images/018_09_crop.png
16% D:/BBDD/018/images/018_10_crop.png
17% D:/BBDD/018/images/018_12_crop.png
17% D:/BBDD/019/images/019_02_crop.png
17% D:/BBDD/019/images/019_03_crop.png
17% D:/BBDD/019/images/019_04_crop.png
18% D:/BBDD/019/images/019_07_crop.png
18% D:/BBDD/019/images/019_08_crop.png
18% D:/BBDD/019/images/019_10_crop.png
18% D:/BBDD/019/images/019_11_crop.png
18% D:/BBDD/019/images/019_12_crop.png
19% D:/BBDD/020/images/020_01_crop.png
19% D:/BBDD/020/images/020_02_crop.png
19% D:/BBDD/020/images/020_04_crop.png
19% D:/BBDD/020/images/020_06_crop.png
19% D:/BBDD/020/images/020_08_crop.png
20% D:/BBDD/020/images/020_09_crop.png
20% D:/BBDD/020/images/020_11_crop.png
20% D:/BBDD/020/images/020_12_crop.png
20% D:/BBDD/021/images/021_01_crop.png
20% D:/BBDD/021/images/021_03_crop.png
21% D:/BBDD/021/images/021_05_crop.png
21% D:/BBDD/021/images/021_07_crop.png
21% D:/BBDD/021/images/021_08_crop.png
21% D:/BBDD/021/images/021_09_crop.png
21% D:/BBDD/021/images/021_11_crop.png
22% D:/BBDD/021/images/021_12_crop.png
22% D:/BBDD/022/images/022_01_crop.png
22% D:/BBDD/022/images/022_03_crop.png
22% D:/BBDD/022/images/022_07_crop.png
23% D:/BBDD/022/images/022_08_crop.png
23% D:/BBDD/022/images/022_09_crop.png
23% D:/BBDD/022/images/022_10_crop.png
23% D:/BBDD/022/images/022_11_crop.png
23% D:/BBDD/022/images/022_12_crop.png
24% D:/BBDD/024/images/024_02_crop.png
24% D:/BBDD/024/images/024_04_crop.png
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

```
24% D:/BBDD/024/images/024_05_crop.png
24% D:/BBDD/024/images/024_06_crop.png
24% D:/BBDD/024/images/024_09_crop.png
25% D:/BBDD/024/images/024_10_crop.png
25% D:/BBDD/024/images/024_11_crop.png
25% D:/BBDD/024/images/024_12_crop.png
25% D:/BBDD/025/images/025_01_crop.png
25% D:/BBDD/025/images/025_02_crop.png
26% D:/BBDD/025/images/025_04_crop.png
26% D:/BBDD/025/images/025_05_crop.png
26% D:/BBDD/025/images/025_07_crop.png
26% D:/BBDD/025/images/025_08_crop.png
26% D:/BBDD/025/images/025_09_crop.png
27% D:/BBDD/025/images/025_10_crop.png
27% D:/BBDD/029/images/029_01_crop.png
27% D:/BBDD/029/images/029_02_crop.png
27% D:/BBDD/029/images/029_04_crop.png
28% D:/BBDD/029/images/029_05_crop.png
28% D:/BBDD/029/images/029_08_crop.png
28% D:/BBDD/029/images/029_10_crop.png
28% D:/BBDD/029/images/029_11_crop.png
28% D:/BBDD/029/images/029_12_crop.png
29% D:/BBDD/030/images/030_02_crop.png
29% D:/BBDD/030/images/030_03_crop.png
29% D:/BBDD/030/images/030_04_crop.png
29% D:/BBDD/030/images/030_05_crop.png
29% D:/BBDD/030/images/030_06_crop.png
30% D:/BBDD/030/images/030_08_crop.png
30% D:/BBDD/030/images/030_10_crop.png
30% D:/BBDD/030/images/030_12_crop.png
30% D:/BBDD/031/images/031_01_crop.png
30% D:/BBDD/031/images/031_02_crop.png
31% D:/BBDD/031/images/031_05_crop.png
31% D:/BBDD/031/images/031_07_crop.png
31% D:/BBDD/031/images/031_09_crop.png
31% D:/BBDD/031/images/031_10_crop.png
31% D:/BBDD/031/images/031_11_crop.png
32% D:/BBDD/031/images/031_12_crop.png
32% D:/BBDD/036/images/036_02_crop.png
32% D:/BBDD/036/images/036_03_crop.png
32% D:/BBDD/036/images/036_04_crop.png
33% D:/BBDD/036/images/036_05_crop.png
33% D:/BBDD/036/images/036_06_crop.png
33% D:/BBDD/036/images/036_08_crop.png
33% D:/BBDD/036/images/036_10_crop.png
33% D:/BBDD/036/images/036_12_crop.png
34% D:/BBDD/037/images/037_01_crop.png
34% D:/BBDD/037/images/037_03_crop.png
34% D:/BBDD/037/images/037_06_crop.png
34% D:/BBDD/037/images/037_07_crop.png
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

```
34% D:/BBDD/037/images/037_08_crop.png
35% D:/BBDD/037/images/037_09_crop.png
35% D:/BBDD/037/images/037_10_crop.png
35% D:/BBDD/037/images/037_11_crop.png
35% D:/BBDD/040/images/040_01_crop.png
35% D:/BBDD/040/images/040_02_crop.png
36% D:/BBDD/040/images/040_03_crop.png
36% D:/BBDD/040/images/040_04_crop.png
36% D:/BBDD/040/images/040_05_crop.png
36% D:/BBDD/040/images/040_10_crop.png
36% D:/BBDD/040/images/040_11_crop.png
37% D:/BBDD/040/images/040_12_crop.png
37% D:/BBDD/042/images/042_01_crop.png
37% D:/BBDD/042/images/042_03_crop.png
37% D:/BBDD/042/images/042_04_crop.png
38% D:/BBDD/042/images/042_06_crop.png
38% D:/BBDD/042/images/042_07_crop.png
38% D:/BBDD/042/images/042_08_crop.png
38% D:/BBDD/042/images/042_10_crop.png
38% D:/BBDD/042/images/042_11_crop.png
39% D:/BBDD/044/images/044_01_crop.png
39% D:/BBDD/044/images/044_04_crop.png
39% D:/BBDD/044/images/044_06_crop.png
39% D:/BBDD/044/images/044_07_crop.png
39% D:/BBDD/044/images/044_08_crop.png
40% D:/BBDD/044/images/044_09_crop.png
40% D:/BBDD/044/images/044_10_crop.png
40% D:/BBDD/044/images/044_12_crop.png
40% D:/BBDD/047/images/047_01_crop.png
40% D:/BBDD/047/images/047_03_crop.png
41% D:/BBDD/047/images/047_04_crop.png
41% D:/BBDD/047/images/047_05_crop.png
41% D:/BBDD/047/images/047_07_crop.png
41% D:/BBDD/047/images/047_08_crop.png
41% D:/BBDD/047/images/047_10_crop.png
42% D:/BBDD/047/images/047_12_crop.png
42% D:/BBDD/049/images/049_01_crop.png
42% D:/BBDD/049/images/049_02_crop.png
42% D:/BBDD/049/images/049_03_crop.png
43% D:/BBDD/049/images/049_04_crop.png
43% D:/BBDD/049/images/049_05_crop.png
43% D:/BBDD/049/images/049_08_crop.png
43% D:/BBDD/049/images/049_09_crop.png
43% D:/BBDD/049/images/049_10_crop.png
44% D:/BBDD/050/images/050_01_crop.png
44% D:/BBDD/050/images/050_04_crop.png
44% D:/BBDD/050/images/050_05_crop.png
44% D:/BBDD/050/images/050_06_crop.png
44% D:/BBDD/050/images/050_07_crop.png
45% D:/BBDD/050/images/050_08_crop.png
```


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

```
45% D:/BBDD/050/images/050_09_crop.png
45% D:/BBDD/050/images/050_10_crop.png
45% D:/BBDD/051/images/051_01_crop.png
45% D:/BBDD/051/images/051_02_crop.png
46% D:/BBDD/051/images/051_03_crop.png
46% D:/BBDD/051/images/051_04_crop.png
46% D:/BBDD/051/images/051_08_crop.png
46% D:/BBDD/051/images/051_09_crop.png
46% D:/BBDD/051/images/051_11_crop.png
47% D:/BBDD/051/images/051_12_crop.png
47% D:/BBDD/052/images/052_01_crop.png
47% D:/BBDD/052/images/052_03_crop.png
47% D:/BBDD/052/images/052_05_crop.png
48% D:/BBDD/052/images/052_06_crop.png
48% D:/BBDD/052/images/052_07_crop.png
48% D:/BBDD/052/images/052_09_crop.png
48% D:/BBDD/052/images/052_11_crop.png
48% D:/BBDD/052/images/052_12_crop.png
49% D:/BBDD/053/images/053_01_crop.png
49% D:/BBDD/053/images/053_04_crop.png
49% D:/BBDD/053/images/053_05_crop.png
49% D:/BBDD/053/images/053_06_crop.png
49% D:/BBDD/053/images/053_08_crop.png
50% D:/BBDD/053/images/053_09_crop.png
50% D:/BBDD/053/images/053_10_crop.png
50% D:/BBDD/053/images/053_12_crop.png
50% D:/BBDD/055/images/055_01_crop.png
50% D:/BBDD/055/images/055_05_crop.png
51% D:/BBDD/055/images/055_06_crop.png
51% D:/BBDD/055/images/055_07_crop.png
51% D:/BBDD/055/images/055_09_crop.png
51% D:/BBDD/055/images/055_10_crop.png
51% D:/BBDD/055/images/055_11_crop.png
52% D:/BBDD/055/images/055_12_crop.png
52% D:/BBDD/056/images/056_02_crop.png
52% D:/BBDD/056/images/056_04_crop.png
52% D:/BBDD/056/images/056_05_crop.png
53% D:/BBDD/056/images/056_06_crop.png
53% D:/BBDD/056/images/056_08_crop.png
53% D:/BBDD/056/images/056_09_crop.png
53% D:/BBDD/056/images/056_10_crop.png
53% D:/BBDD/056/images/056_12_crop.png
54% D:/BBDD/060/images/060_01_crop.png
54% D:/BBDD/060/images/060_02_crop.png
54% D:/BBDD/060/images/060_04_crop.png
54% D:/BBDD/060/images/060_06_crop.png
54% D:/BBDD/060/images/060_07_crop.png
55% D:/BBDD/060/images/060_08_crop.png
55% D:/BBDD/060/images/060_10_crop.png
55% D:/BBDD/060/images/060_12_crop.png
```

```
55% D:/BBDD/061/images/061_01_crop.png
55% D:/BBDD/061/images/061_03_crop.png
56% D:/BBDD/061/images/061_04_crop.png
56% D:/BBDD/061/images/061_05_crop.png
56% D:/BBDD/061/images/061_07_crop.png
56% D:/BBDD/061/images/061_08_crop.png
56% D:/BBDD/061/images/061_10_crop.png
57% D:/BBDD/061/images/061_11_crop.png
57% D:/BBDD/062/images/062_01_crop.png
57% D:/BBDD/062/images/062_03_crop.png
57% D:/BBDD/062/images/062_04_crop.png
58% D:/BBDD/062/images/062_05_crop.png
58% D:/BBDD/062/images/062_07_crop.png
58% D:/BBDD/062/images/062_08_crop.png
58% D:/BBDD/062/images/062_10_crop.png
58% D:/BBDD/062/images/062_11_crop.png
59% D:/BBDD/067/images/067_01_crop.png
59% D:/BBDD/067/images/067_02_crop.png
59% D:/BBDD/067/images/067_03_crop.png
59% D:/BBDD/067/images/067_05_crop.png
59% D:/BBDD/067/images/067_06_crop.png
60% D:/BBDD/067/images/067_07_crop.png
60% D:/BBDD/067/images/067_11_crop.png
60% D:/BBDD/067/images/067_12_crop.png
60% D:/BBDD/068/images/068_02_crop.png
60% D:/BBDD/068/images/068_03_crop.png
61% D:/BBDD/068/images/068_04_crop.png
61% D:/BBDD/068/images/068_05_crop.png
61% D:/BBDD/068/images/068_06_crop.png
61% D:/BBDD/068/images/068_07_crop.png
61% D:/BBDD/068/images/068_11_crop.png
62% D:/BBDD/068/images/068_12_crop.png
62% D:/BBDD/070/images/070_01_crop.png
62% D:/BBDD/070/images/070_02_crop.png
62% D:/BBDD/070/images/070_03_crop.png
63% D:/BBDD/070/images/070_07_crop.png
63% D:/BBDD/070/images/070_08_crop.png
63% D:/BBDD/070/images/070_09_crop.png
63% D:/BBDD/070/images/070_10_crop.png
63% D:/BBDD/070/images/070_12_crop.png
64% D:/BBDD/071/images/071_03_crop.png
64% D:/BBDD/071/images/071_04_crop.png
64% D:/BBDD/071/images/071_05_crop.png
64% D:/BBDD/071/images/071_06_crop.png
64% D:/BBDD/071/images/071_07_crop.png
65% D:/BBDD/071/images/071_10_crop.png
65% D:/BBDD/071/images/071_11_crop.png
65% D:/BBDD/071/images/071_12_crop.png
65% D:/BBDD/072/images/072_01_crop.png
65% D:/BBDD/072/images/072_02_crop.png
```

```
66% D:/BBDD/072/images/072_03_crop.png
66% D:/BBDD/072/images/072_04_crop.png
66% D:/BBDD/072/images/072_05_crop.png
66% D:/BBDD/072/images/072_07_crop.png
66% D:/BBDD/072/images/072_08_crop.png
67% D:/BBDD/072/images/072_11_crop.png
67% D:/BBDD/074/images/074_01_crop.png
67% D:/BBDD/074/images/074_02_crop.png
67% D:/BBDD/074/images/074_04_crop.png
68% D:/BBDD/074/images/074_05_crop.png
68% D:/BBDD/074/images/074_06_crop.png
68% D:/BBDD/074/images/074_08_crop.png
68% D:/BBDD/074/images/074_09_crop.png
68% D:/BBDD/074/images/074_12_crop.png
69% D:/BBDD/075/images/075_02_crop.png
69% D:/BBDD/075/images/075_04_crop.png
69% D:/BBDD/075/images/075_05_crop.png
69% D:/BBDD/075/images/075_06_crop.png
69% D:/BBDD/075/images/075_08_crop.png
70% D:/BBDD/075/images/075_09_crop.png
70% D:/BBDD/075/images/075_11_crop.png
70% D:/BBDD/075/images/075_12_crop.png
70% D:/BBDD/076/images/076_01_crop.png
70% D:/BBDD/076/images/076_02_crop.png
71% D:/BBDD/076/images/076_03_crop.png
71% D:/BBDD/076/images/076_06_crop.png
71% D:/BBDD/076/images/076_08_crop.png
71% D:/BBDD/076/images/076_09_crop.png
71% D:/BBDD/076/images/076_10_crop.png
72% D:/BBDD/076/images/076_12_crop.png
72% D:/BBDD/078/images/078_01_crop.png
72% D:/BBDD/078/images/078_02_crop.png
72% D:/BBDD/078/images/078_03_crop.png
73% D:/BBDD/078/images/078_04_crop.png
73% D:/BBDD/078/images/078_06_crop.png
73% D:/BBDD/078/images/078_07_crop.png
73% D:/BBDD/078/images/078_08_crop.png
73% D:/BBDD/078/images/078_10_crop.png
74% D:/BBDD/079/images/079_01_crop.png
74% D:/BBDD/079/images/079_02_crop.png
74% D:/BBDD/079/images/079_06_crop.png
74% D:/BBDD/079/images/079_08_crop.png
74% D:/BBDD/079/images/079_09_crop.png
75% D:/BBDD/079/images/079_10_crop.png
75% D:/BBDD/079/images/079_11_crop.png
75% D:/BBDD/079/images/079_12_crop.png
75% D:/BBDD/081/images/081_01_crop.png
75% D:/BBDD/081/images/081_02_crop.png
76% D:/BBDD/081/images/081_03_crop.png
76% D:/BBDD/081/images/081_04_crop.png
```

```
76% D:/BBDD/081/images/081_05_crop.png
76% D:/BBDD/081/images/081_07_crop.png
76% D:/BBDD/081/images/081_08_crop.png
77% D:/BBDD/081/images/081_12_crop.png
77% D:/BBDD/082/images/082_01_crop.png
77% D:/BBDD/082/images/082_02_crop.png
77% D:/BBDD/082/images/082_04_crop.png
78% D:/BBDD/082/images/082_06_crop.png
78% D:/BBDD/082/images/082_09_crop.png
78% D:/BBDD/082/images/082_10_crop.png
78% D:/BBDD/082/images/082_11_crop.png
78% D:/BBDD/082/images/082_12_crop.png
79% D:/BBDD/083/images/083_01_crop.png
79% D:/BBDD/083/images/083_02_crop.png
79% D:/BBDD/083/images/083_03_crop.png
79% D:/BBDD/083/images/083_04_crop.png
79% D:/BBDD/083/images/083_08_crop.png
80% D:/BBDD/083/images/083_09_crop.png
80% D:/BBDD/083/images/083_10_crop.png
80% D:/BBDD/083/images/083_12_crop.png
80% D:/BBDD/085/images/085_01_crop.png
80% D:/BBDD/085/images/085_03_crop.png
81% D:/BBDD/085/images/085_07_crop.png
81% D:/BBDD/085/images/085_08_crop.png
81% D:/BBDD/085/images/085_09_crop.png
81% D:/BBDD/085/images/085_10_crop.png
81% D:/BBDD/085/images/085_11_crop.png
82% D:/BBDD/085/images/085_12_crop.png
82% D:/BBDD/088/images/088_02_crop.png
82% D:/BBDD/088/images/088_03_crop.png
82% D:/BBDD/088/images/088_05_crop.png
83% D:/BBDD/088/images/088_06_crop.png
83% D:/BBDD/088/images/088_07_crop.png
83% D:/BBDD/088/images/088_09_crop.png
83% D:/BBDD/088/images/088_10_crop.png
83% D:/BBDD/088/images/088_12_crop.png
84% D:/BBDD/090/images/090_01_crop.png
84% D:/BBDD/090/images/090_02_crop.png
84% D:/BBDD/090/images/090_04_crop.png
84% D:/BBDD/090/images/090_06_crop.png
84% D:/BBDD/090/images/090_08_crop.png
85% D:/BBDD/090/images/090_09_crop.png
85% D:/BBDD/090/images/090_11_crop.png
85% D:/BBDD/090/images/090_12_crop.png
85% D:/BBDD/091/images/091_01_crop.png
85% D:/BBDD/091/images/091_02_crop.png
86% D:/BBDD/091/images/091_03_crop.png
86% D:/BBDD/091/images/091_05_crop.png
86% D:/BBDD/091/images/091_06_crop.png
86% D:/BBDD/091/images/091_09_crop.png
```

```
86% D:/BBDD/091/images/091_10_crop.png
87% D:/BBDD/091/images/091_11_crop.png
87% D:/BBDD/093/images/093_01_crop.png
87% D:/BBDD/093/images/093_04_crop.png
87% D:/BBDD/093/images/093_05_crop.png
88% D:/BBDD/093/images/093_06_crop.png
88% D:/BBDD/093/images/093_07_crop.png
88% D:/BBDD/093/images/093_08_crop.png
88% D:/BBDD/093/images/093_09_crop.png
88% D:/BBDD/093/images/093_10_crop.png
89% D:/BBDD/095/images/095_01_crop.png
89% D:/BBDD/095/images/095_02_crop.png
89% D:/BBDD/095/images/095_04_crop.png
89% D:/BBDD/095/images/095_06_crop.png
89% D:/BBDD/095/images/095_07_crop.png
90% D:/BBDD/095/images/095_08_crop.png
90% D:/BBDD/095/images/095_09_crop.png
90% D:/BBDD/095/images/095_10_crop.png
90% D:/BBDD/096/images/096_02_crop.png
90% D:/BBDD/096/images/096_04_crop.png
91% D:/BBDD/096/images/096_05_crop.png
91% D:/BBDD/096/images/096_07_crop.png
91% D:/BBDD/096/images/096_08_crop.png
91% D:/BBDD/096/images/096_09_crop.png
91% D:/BBDD/096/images/096_10_crop.png
92% D:/BBDD/096/images/096_11_crop.png
92% D:/BBDD/097/images/097_01_crop.png
92% D:/BBDD/097/images/097_02_crop.png
92% D:/BBDD/097/images/097_03_crop.png
93% D:/BBDD/097/images/097_04_crop.png
93% D:/BBDD/097/images/097_05_crop.png
93% D:/BBDD/097/images/097_07_crop.png
93% D:/BBDD/097/images/097_09_crop.png
93% D:/BBDD/097/images/097_12_crop.png
94% D:/BBDD/098/images/098_01_crop.png
94% D:/BBDD/098/images/098_02_crop.png
94% D:/BBDD/098/images/098_03_crop.png
94% D:/BBDD/098/images/098_04_crop.png
94% D:/BBDD/098/images/098_05_crop.png
95% D:/BBDD/098/images/098_08_crop.png
95% D:/BBDD/098/images/098_09_crop.png
95% D:/BBDD/098/images/098_11_crop.png
95% D:/BBDD/100/images/100_01_crop.png
95% D:/BBDD/100/images/100_03_crop.png
96% D:/BBDD/100/images/100_04_crop.png
96% D:/BBDD/100/images/100_07_crop.png
96% D:/BBDD/100/images/100_08_crop.png
96% D:/BBDD/100/images/100_10_crop.png
96% D:/BBDD/100/images/100_11_crop.png
97% D:/BBDD/100/images/100_12_crop.png
```

```
 97% D:/BBDD/102/images/102_02_crop.png
 97% D:/BBDD/102/images/102_03_crop.png
 97% D:/BBDD/102/images/102_04_crop.png
 98% D:/BBDD/102/images/102_06_crop.png
 98% D:/BBDD/102/images/102_08_crop.png
 98% D:/BBDD/102/images/102_09_crop.png
 98% D:/BBDD/102/images/102_10_crop.png
 98% D:/BBDD/102/images/102_11_crop.png
 99% D:/BBDD/103/images/103_01_crop.png
 99% D:/BBDD/103/images/103_02_crop.png
 99% D:/BBDD/103/images/103_03_crop.png
 99% D:/BBDD/103/images/103_04_crop.png
 99% D:/BBDD/103/images/103_07_crop.png
100% D:/BBDD/103/images/103_08_crop.png
100% D:/BBDD/103/images/103_11_crop.png
100% D:/BBDD/103/images/103_12_crop.png
Loaded 480 files. 480 of 480 were valid
Loading data ended in 24.0078 seconds
Creating aam... Ended in 91.0696 seconds
Saving aam... Ended in 6.4931 seconds
```

Apéndice E

Uso interfaz gráfica (GUI)

Nos ha parecido interesante desarrollar una interfaz gráfica para poder utilizar los programas creados de una manera sencilla y didáctica. No se ha implementado la fase de optimización, ya que no es necesaria al haberse realizado ya el estudio de cuales son los AAM óptimos para cada método.
La interfaz se compone de 4 programas distintos (cada uno accesible desde el menú correspondiente):

1. AAM: hay información sobre en que consiste AAM y la ayuda para poder utilizar el resto del programa.
2. BBDD: están presentes todas las opciones referidas a al base de datos de imágenes. Importar los ficheros necesarios de la base de datos GI4E, exportar los ficheros adicionales y el borrado de los mismos en caso de necesidad..
3. Training: nos permite obtener un fichero AAM con las especificaciones deseadas.
4. Testing: nos permite poner a prueba los ficheros AAM generados en la fase de entrenamiento sobre las imágenes de la base de datos.

E.1. AAM

Para acceder a este apartado hay que seleccionarlo en el menú AAM.
En el apartado apartado "Home" está disponible la ayuda y en el apartado "AAM" hay un acceso a un PDF con información sobre AAM.

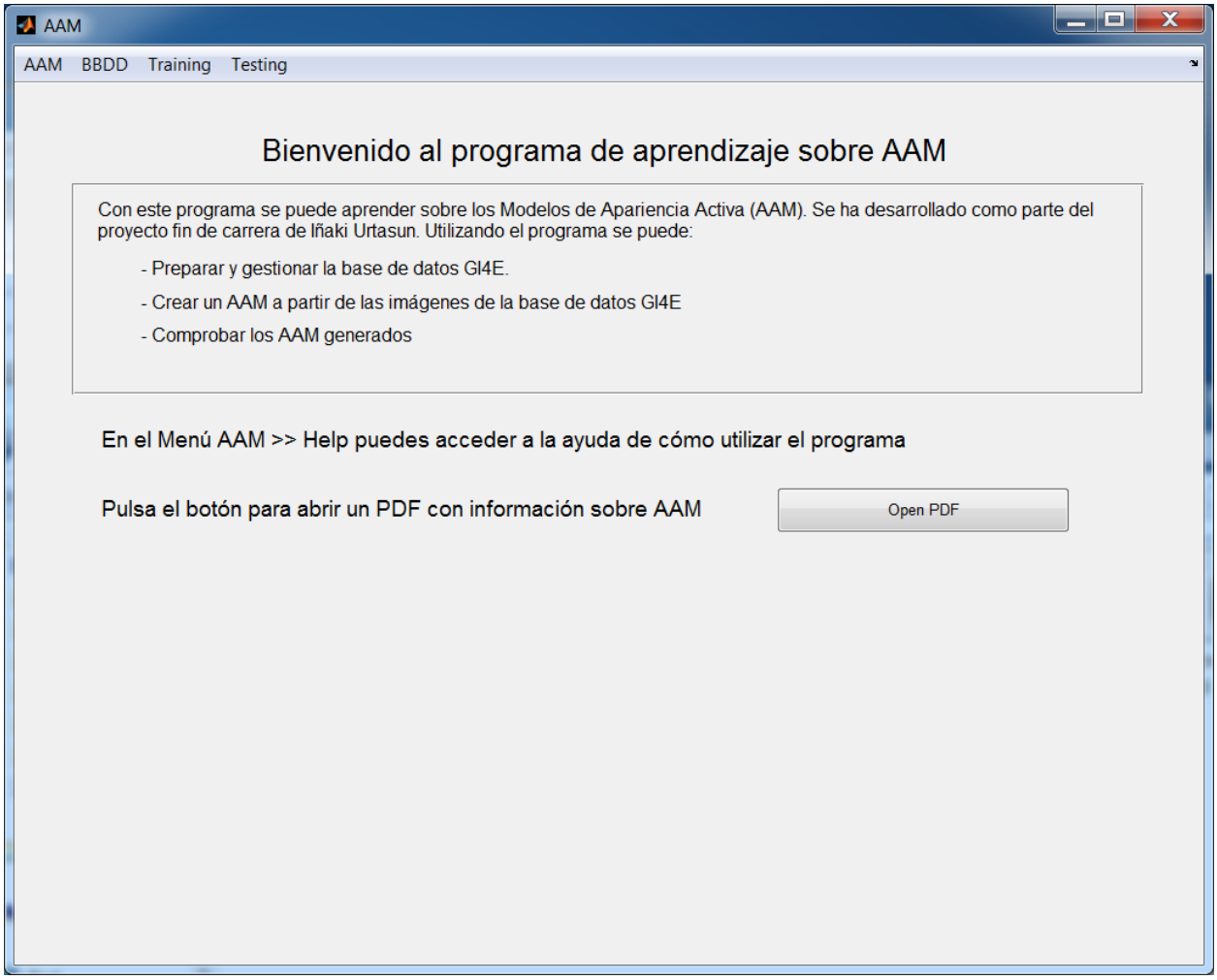


Figura E.1: GUI AAM

E.2. BBDD

Para acceder a este apartado hay que seleccionarlo desde el menú BBDD.
Una vez ya tengamos cargado el programa aparecerá la pantalla que se puede observar en la figura E.2. Podemos especificar en los campos correspondientes las rutas necesarias:

- GI4E Database folder: ruta de la ubicación de la base de datos GI4E.
- Database folder: ruta de la ubicación deseada para la base de datos con los datos imprescindibles.

Una vez especificadas las rutas podemos pulsar el botón "Import G4ie DDBB" para que comience la importación. El proceso durará unos minutos. En la consola del programa MATLAB® se puede ver el proceso de manera más detallada.
Una vez terminada la importación se activará el botón "Create files" para crear los archivos complementarios (ficheros de máscara, landmarks separados, imágenes recortadas...). Púlsalo para que comience le proceso. Como en el caso anterior, en la consola del programa MATLAB® se puede ver el proceso de manera más detallada.
En caso de que fuese necesario, pulsando el botón "Clear files", se procede al borrado de los ficheros creados en el proceso explicado en el párrafo anterior.
Se recomienda crear los ficheros complementarios antes de continuar con las fases de entrenamiento y testeo.

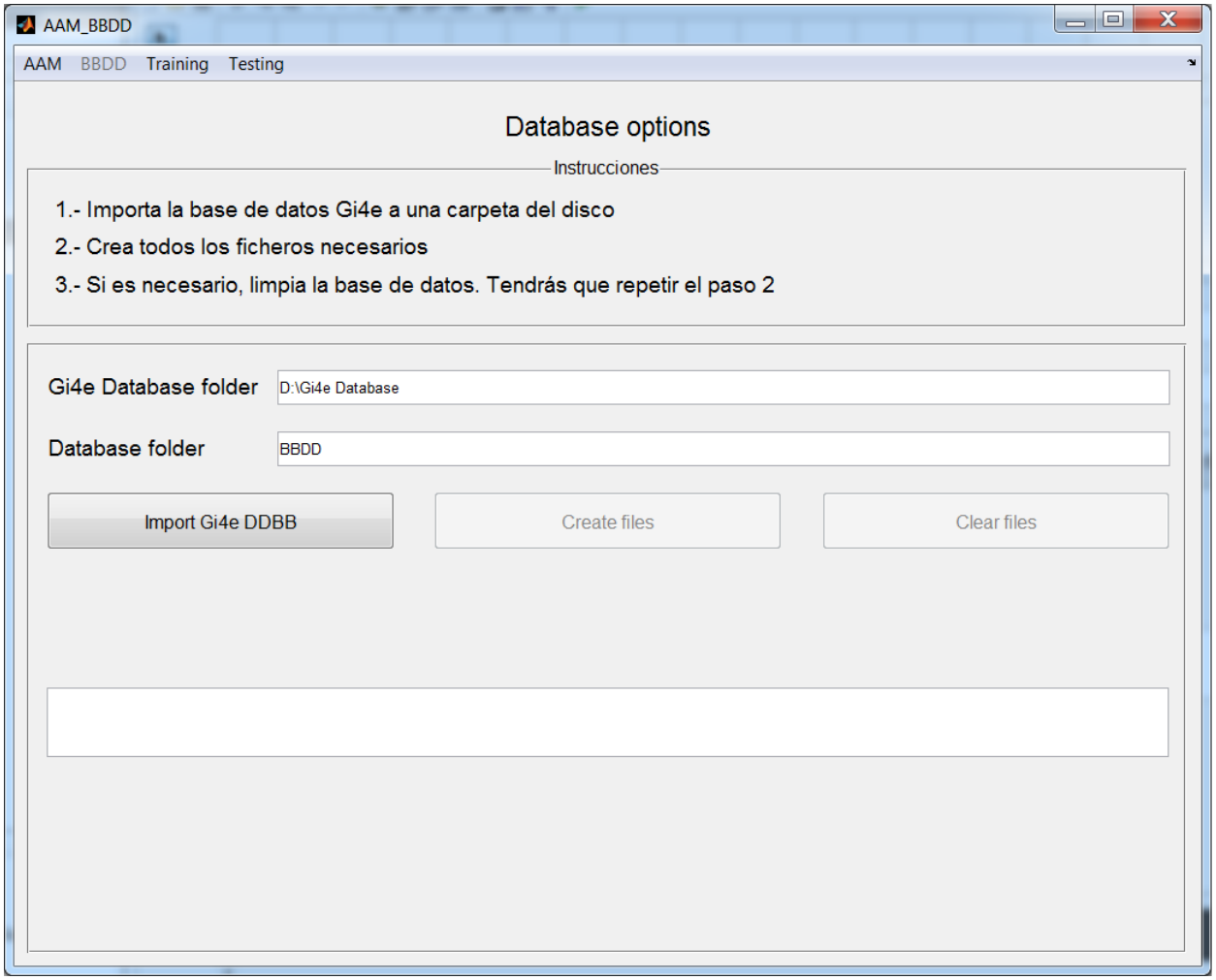


Figura E.2: GUI BBDD

E.3. Fase de entrenamiento

Para acceder a este apartado hay que seleccionarlo desde el menú Training.
Una vez ya tengamos cargado el programa aparecerá la pantalla que se puede observar en la figura E.3. Las distintas opciones disponibles se dividen en tres apartados principales.
Referidas a la base de datos:

- DDBB folder: control editable donde se especifica la ubicación de la base de datos.
- Images per user: control deslizable donde se puede escoger el número de imágenes que se escogerán de cada usuario. Valores disponibles: entre 1 y 12.
- Users for training: control deslizable donde se puede escoger el número de usuarios distintos que se escogerán para crear el AAM. Valores disponibles: entre 40 y 70.
- Use crop images: control checkbox que indica si utilizamos las imágenes recortadas o las originales.
- Use crop images: control checkbox que indica si se exportarán o no los ficheros complementarios que se vayan creando en el proceso.
- Clear user files: control checkbox que indica si se limpiarán los ficheros complementarios de un usuario antes de comenzar el proceso.

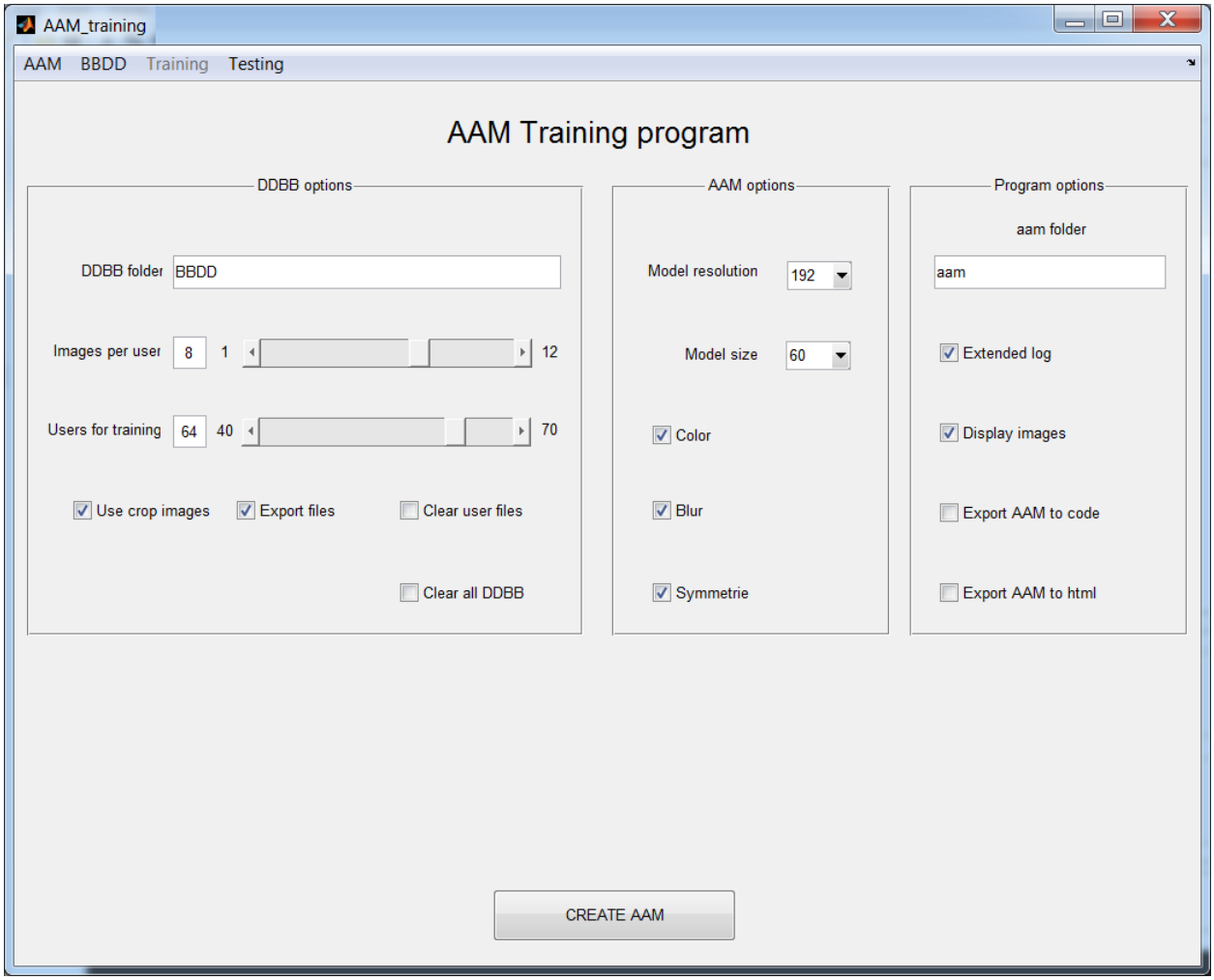


Figura E.3: GUI training

- Clear all DDBB: control checkbox que indica si se limpiarán todos los ficheros complementarios de todos los usuarios antes de comenzar el proceso.

Referidas a las opciones de creación de AAM:

- Model resolution: control desplegable donde se especifica la resolución del modelo. Valores disponibles 128, 192 y 256.
- Model size: control desplegable donde se puede escoger el número de imágenes que integrarán el modelo. Valores disponibles 60, 70, 80, 90 y 100.
- Color: control checkbox que indica si el AAM creado será en color o no.
- Blur: control checkbox que indica si se activa o no el desenfoque.
- Symmetrie: control checkbox que indica si utiliza la simetría especular para duplicar el número de imágenes de entrenamiento.

Referidas a las opciones del programa:

- aam folder: control editable donde se especifica la carpeta donde se guardará el fichero AAM.
- Extended log: control checkbox que indica si se crea un log detallado de todo el proceso de creación.

- Display images: control checkbox que indica si se muestran algunas de las imágenes utilizadas en la creación del fichero AAM.
- Export AAM to code: control checkbox que indica si se exportará el AAM generado al formato necesario para su uso mediante la librería creada en C++ (libaam_code).
- Export AAM to html: control checkbox que indica si se exportará el AAM generado a html para visualizar su contenido mediante un navegador web.

Una vez escogidos los parámetros a nuestro gusto, hay que pulsar el botón "CREATE AAM" para que se genere el fichero AAM. Este proceso puede costar varios minutos y necesita gran cantidad de memoria, por lo que se recomienda cerrar el resto de programas que pudiesen estar activos. Una vez finalizado, en la carpeta especificada, dispondremos del fichero AAM. Se puede repetir el proceso tantas veces como se quiera para generar distintos AAM. El nombre del AAM generado depende de los distintos parámetros escogidos, por lo que si se ponen exactamente los mismos parámetros, se sobrescribirá cualquier AAM que estuviese ya creado con dichas opciones.

E.4. Fase de testeo

Para acceder a este apartado hay que seleccionarlo desde el menú Testing.
Una vez ya tengamos cargado el programa aparecerá la pantalla que se puede observar en la figura E.4. En ese primer momento el único apartado apartado que podemos modificar es:

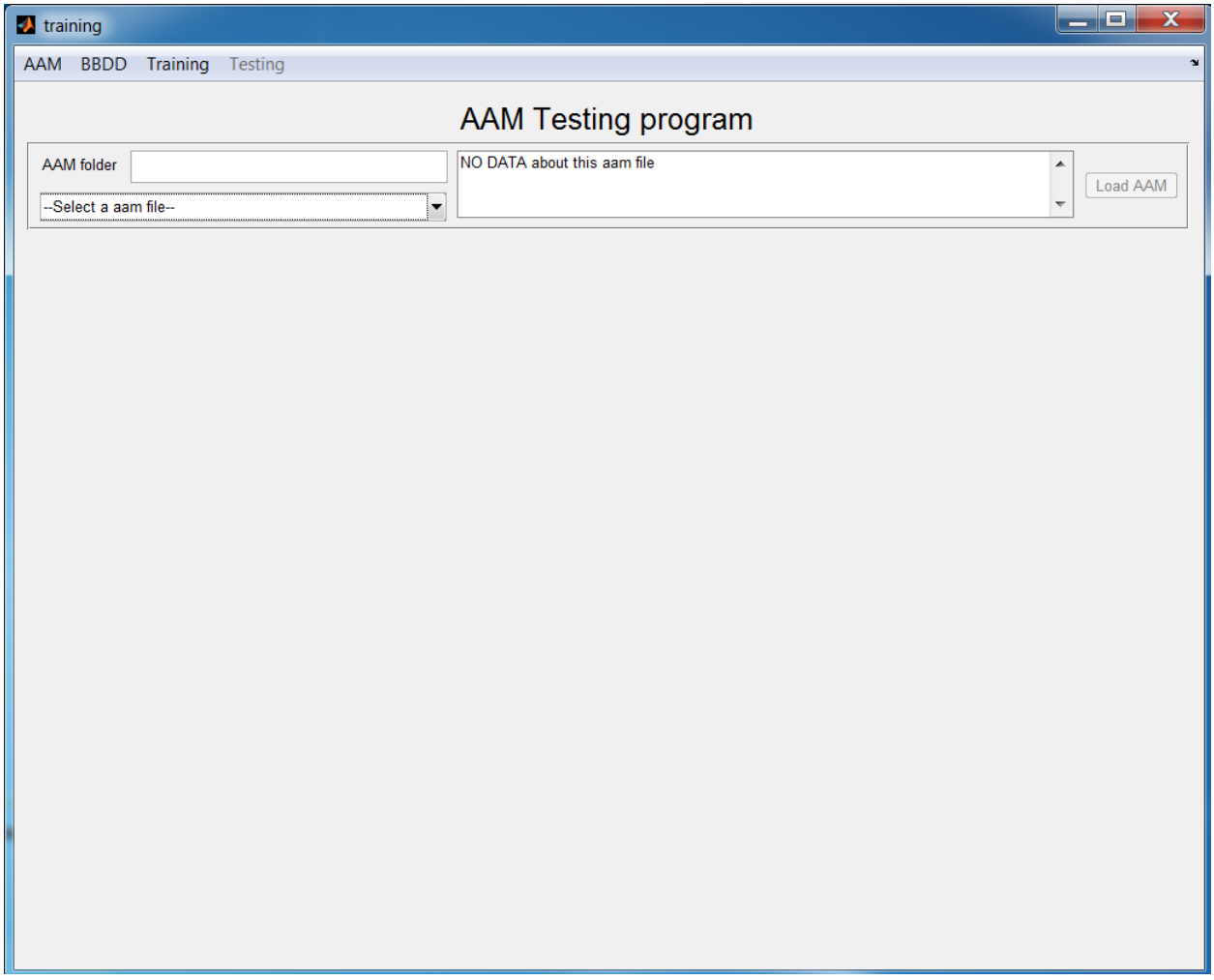


Figura E.4: GUI testing, pantalla inicial

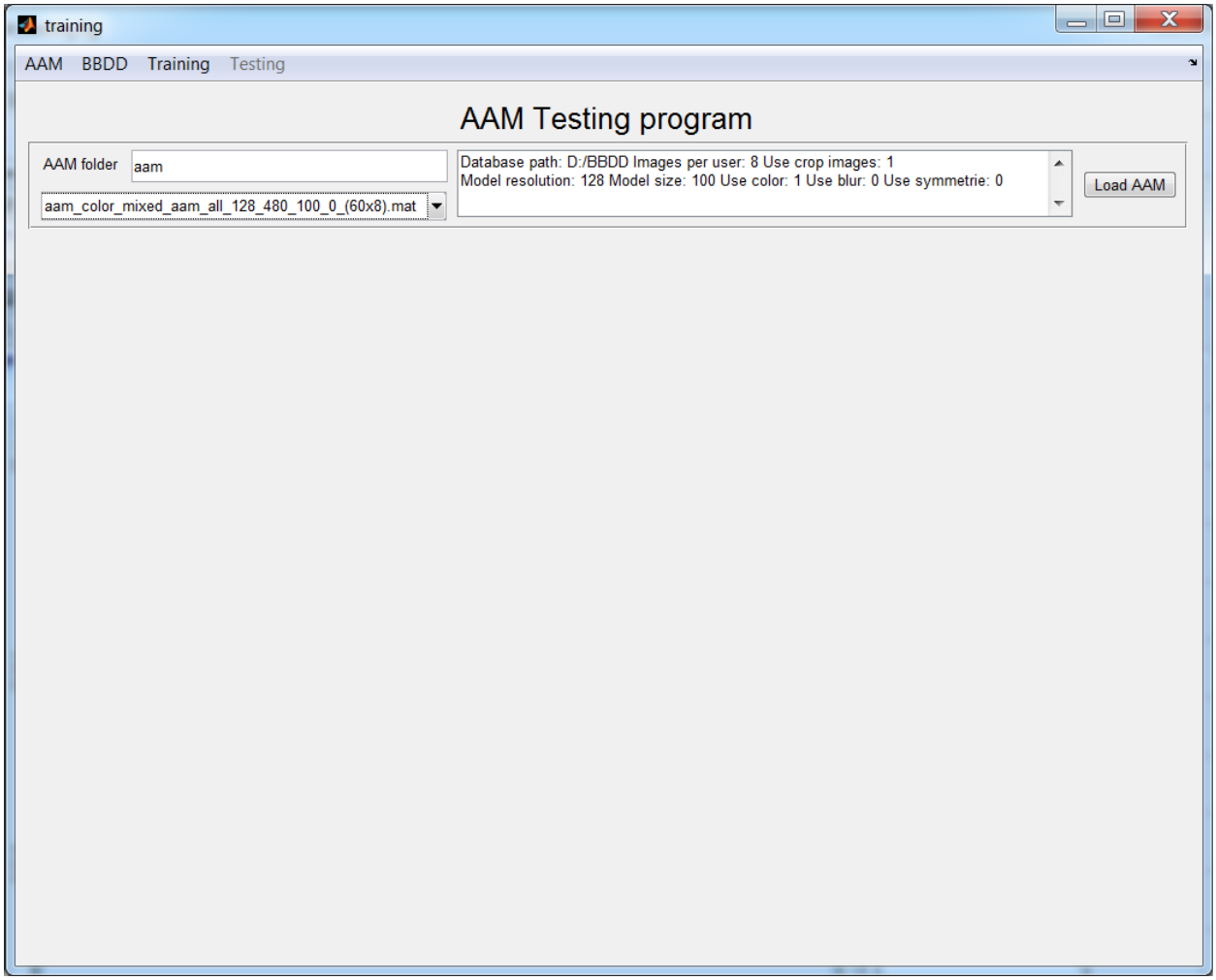


Figura E.5: GUI testing, carga del fichero AAM

- AAM folder: ubicación donde está la carpeta con los ficheros AAM.

Si la dirección proporcionada en "AAM folder" es correcta, y contiene algún fichero AAM en su interior, éste aparecerá en la lista despegable. Cuando seleccionemos uno de ellos nos aparecerá la información disponible sobre el mismo (tal y como se puede observar en la figura E.5) y se activará el botón "Load AAM". Una vez lo pulsemos se cargará el fichero AAM para poder trabajar con él (esto tarda unos segundos).
Una vez cargado el fichero AAM aparecerá el botón "Reset AAM" en el lugar donde estaba localizado el botón que permitía cargar el AAM. Si lo pulsamos volveremos al principio del programa y podremos seleccionar otro AAM distinto.
Además se activan los siguientes controles:

- DDBB folder: control editable donde se especifica la carpeta donde se ubica la base de datos de imágenes.
- Select user: lista editable donde se pueden seleccionar los usuarios disponibles para la fase de testeo. Conviene recordar que la lista de los usuarios para testear está en el interior del propio fichero AAM, ya que no se pueden utilizar aquellos que han intervenido en la fase de entrenamiento.
- Select image: lista editable en el que escogeremos una de las doce imágenes del usuario seleccionado.

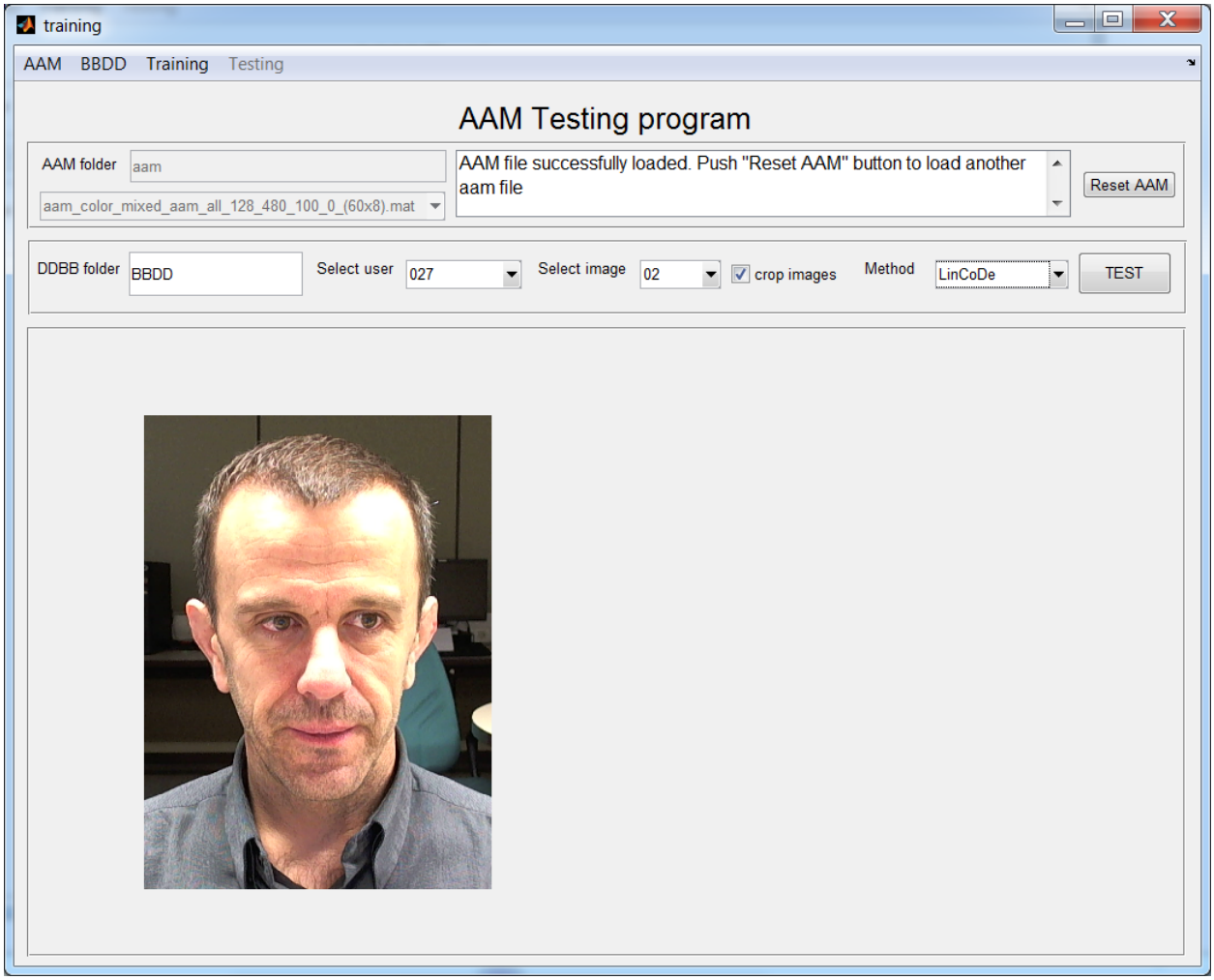


Figura E.6: GUI testing, selección de una imagen de usuario

- crop images: control checkbox que indica si utilizaremos la imagen completa o recortada.
- Method: lista editable en la que podremos escoger uno de los siguientes métodos implementados:
  - LinCoLiNe
  - CoLiNe
  - LinCoDe
  - CoDe
  - CoNe

Una vez seleccionadas las opciones que queramos (ver ejemplo de la figura E.6) podemos pulsar el botón "TEST" que nos ajusta la imagen escogida con el AAM seleccionado y el método elegido, tal como podemos ver en la figura E.7. Cabe destacar que los puntos verdes son los puntos de referencia marcados a mano y los puntos azules son los calculados en el proceso de ajuste

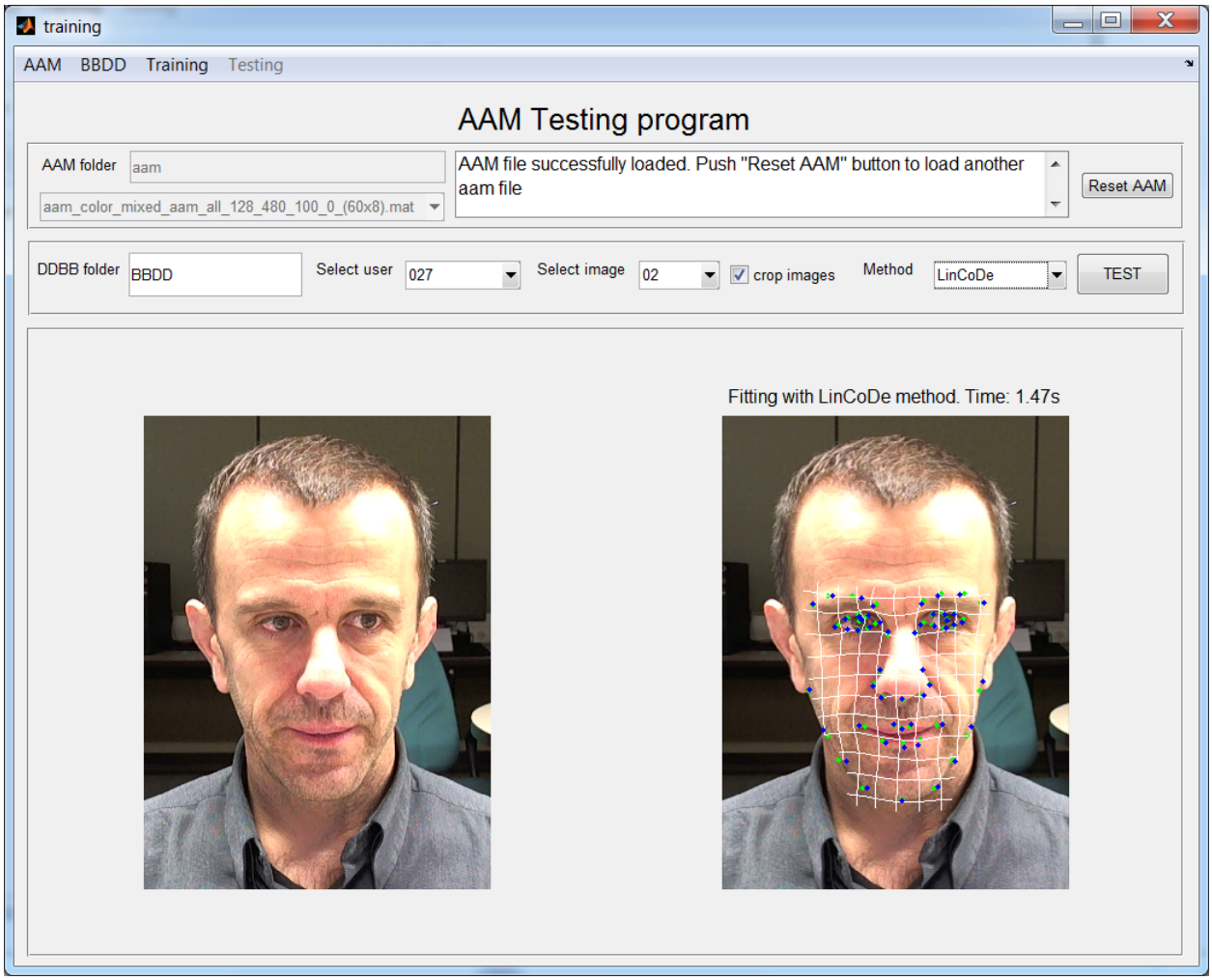


Figura E.7: GUI testing, ajuste por el método LinCoDe sobre la imagen 2 del usuario 027