3.2. SOLUCIÓN QUE OFRECEMOS

Tras estudiar y analizar todas las librerías nuestro objetivo ha sido el de crear una sola base de datos que aúne la mayor variedad posible de tipos de imágenes, y de variedad en la tipología de los sujetos dentro de nuestras posibilidades.

Si analizamos la tabla del apartado anterior podemos ver que únicamente la base de datos PIE trabaja en más campos que Mediterranean Faces.

Los dos ámbitos que nos interesaba más trabajar son, con excepción del de imágenes con luz visible y posadas, el de imágenes infrarrojas y el de imágenes espontaneas. Esto tiene una explicación completamente lógica y es que, como ya hemos mencionado antes, el principal objetivo de la base de datos es satisfacer el entrenamiento de algoritmos que se usen en tareas de vigilancia.

Por ello lo que hemos pretendido es que la librería Mediterranean Faces contuviese imágenes espontaneas, pero con un mínimo de control ya que, de no haber sido así muchas no hubiesen podido ni siquiera ser detectadas por el algoritmo de detección Viola-Jones. Este control ha sido realizado manualmente durante el proceso de recolección de sujetos y de captura de imágenes procesando cada una de las imágenes para comprobar la detección positiva del rostro del sujeto en cuestión.

El otro campo que consideramos de mayor importancia y que diferencia a Mediterranean Faces de la mayor parte de las bases de datos de caras es la inclusión de imágenes infrarrojas, ya que consideramos este apartado fundamental para cualquier base de datos que pretenda orientarse a aplicaciones de vigilancia.

Existen dos librerías más que sí contienen imágenes infrarrojas, estas son la *Natural Visible and Infrared Expression database (NVIE) y Surveillance Camera Faces (SC Faces),* pero no son objeto de análisis en este proyecto de investigación.

Por otra parte hemos querido incluir variedad en los gestos para, en primer lugar añadir un grado de dificultad al reconocimiento y además, pensando en aplicaciones orientadas al reconocimiento gestual.

La dificultad añadida para el reconocimiento resulta obvia si lo analizamos. Las posibilidades de reconocimiento positivo serán mucho más altas si entrenamos el algoritmo con una muestra de imágenes de

gran similitud entre ellas para luego pasarle como imagen de test otra que también sea prácticamente igual.

En cuanto al reconocimiento gestual, es un campo que aun esta por expandirse al nivel del reconocimiento o la detección facial, pero ya encontramos aplicaciones a nivel usuario donde podemos observar la aplicación de este tipo de tecnología. La más conocida tal vez sea la captura de fotos automática al detectar sonrisas, tecnología que ya encontramos en algunas cámaras de fotos desde hace años.

Por ello nuestro objetivo ha sido el de no solo añadir gestos sino el de hacerlo de manera ordenada y estructurada, asi si alguien le interesa trabajar con las imágenes de sonrisa solo tendría que seleccionar de manera manual o automática todas las imágenes con subíndice 2 del directorio *Visible.*

Del mismo modo que con los gestos podemos encontrar dos imágenes por sujeto en las que hemos añadido oclusiones al sujeto tales como gafas, gafas de sol, pañuelos, o sombreros.

4. DISEÑO DE LA LIBRERÍA MEDITERRANEAN FACES

4.1. ESTRUCTURA DE MEDITERRANEAN FACES

Uno de los pasos previos a la captura de imágenes ha sido el estudio del resto de librerías con el fin de analizar su estructura y organización. Al realizar este análisis nos hemos encontrado con que la gran mayoría de bases de datos carecen de una estructura manejable. En la mayoría de los casos se necesita del documento explicativo para poder manejarla con suficiente fluidez. Asimismo encontramos librerías en las que el numero de imágenes por sujeto varia de forma significativa.

Por ello uno de los objetivos que nos planteamos fue el de crear una base de datos ordenada, homogénea y con una estructura simple y de fácil manejo.

La base de datos Mediterranean Faces puede ser fácilmente leída, ya sea total o parcialmente con funciones como las que expondremos en puntos posteriores.

Encontramos tres grupos principales de imágenes donde se almacenan las capturas originales sin procesado alguno:

- Imágenes visibles – Las encontramos en la carpeta “Visible”.

Figura 4.1 –Imagen tipo Visible -

- Imágenes infrarrojas – Las encontramos en la carpeta” IR”.

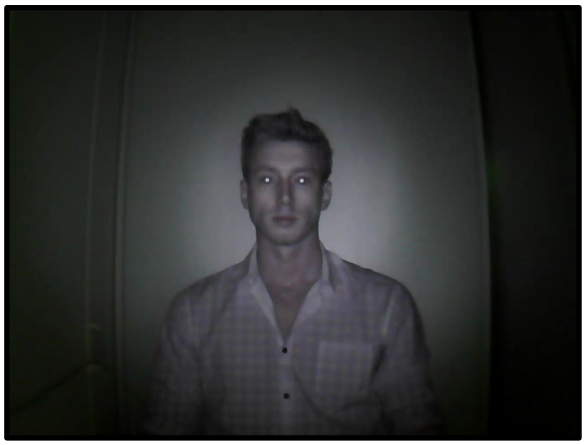

Figura 4.2 –Imagen tipo IR -

- Imágenes no posadas – Las encontramos en la carpeta “Wild”.

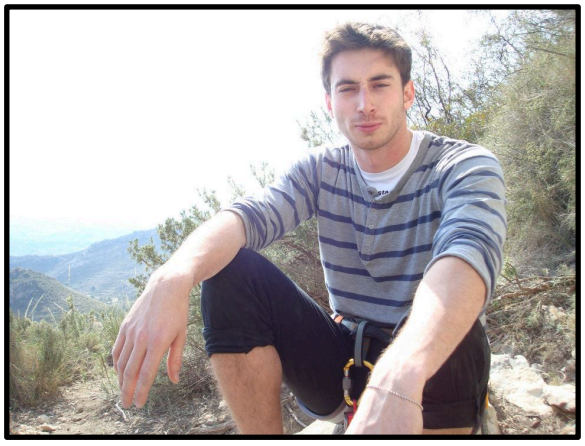

Figura 4.3 –Imagen tipo Wild -

Asimismo existen otras tres carpetas donde las imágenes ya han sido procesadas por la función que incorpora el algoritmo Viola-Jones:

- Imágenes visibles procesadas– Las encontramos en la carpeta "VisibleC".

Figura 4.4 –Imagen tipo Visible *recortada o* VisibleC -

- Imágenes infrarrojas procesadas – Las encontramos en la carpeta" IRC".

Figura 4.6 –Imagen tipo IR *recortda o* IRC -

- Imágenes no posadas procesadas – Las encontramos en la carpeta "WildC"
-

Figura 4.6 –Imagen tipo Wild *recortda o* WildC -

Dentro de cada carpeta hemos optado por no crear subdirectorios, de esta manera facilitamos el manejo de las imágenes con funciones. Las imágenes se nombran siguiendo la siguiente estructura:

s"numero de sujeto"_"numero de imagen".

Por ejemplo, si buscamos la imagen 5 del sujeto 12 su nomenclatura será:

"s12_5".

Esta estructura se sigue para cada una de las tres carpetas principales así como para las tres procesadas.

En el interior de las carpetas procesadas, *VisibleC*, *IRC* y *WildC,* encontramos un subdirectorios llamado coordenadas en el que encontramos un archivo de texto por cada unas de las imágenes y con el mismo nombre que estas.

Este archivo de coordenadas es el resultado de procesar las imágenes con el algoritmo Viola-Jones. Una vez elegimos el tamaño que queremos para la cara recortada el algoritmo guarda una imagen con estas dimensiones.

Supongamos que elegimos recortar una cara en un cuadrado de 100x100 pixeles. Una vez procesamos la imagen, la función nos dará un cuadrado con vértices en [(x1,y1) (x2,y2) (x3,y3) (x4,y4)], para a continuación contraer la imagen al tamaño deseado, en este caso 100x100. Si quisiésemos expandir la imagen a partir de la original también seria posible, pero con la evidente perdida de resolución.

Figura 4.7 –Ejemplo de recorte por algoritmo Viola-Jones –

Figura 4.8 –Ejemplo de cara recortara por el algoritmo Viola-Jones -

En el archivo de coordenadas encontraremos las correspondientes a las esquina superior izquierda y la inferior derecha: (x1,y1) (x3,y3).

Gracias al archivo de coordenadas solo nos haría falta procesar las imágenes una sola vez, ya que, una hemos pasado la función de procesamiento que contiene el algoritmo Viola-Jones las coordenadas son guardadas. Estas coordenadas son las que se obtienen sobre la fotografía original, sin tener en cuenta el tamaño deseado, y si volvemos a necesitar detectar la cara la función acudiría a las coordenadas, y deberíamos introducir el tamaño de recorte deseado, ahorrando de esta manera el tiempo que consume el algoritmo en el análisis de la imagen.

4.2. CREACIÓN DE FUNCIONES DE MATLAB PARA ESTRUCTURAR LA LIBRERÍA

FUNCIÓN VIOLA_VISIBLE.

Utilizamos la función Viola_visible para, directamente desde el directorio donde se guardan las imágenes de la cámara Logitech, procesar las imágenes y obtener una cara de cada sujeto.

La ventaja de esta función es su inmediatez, aunque, debido a las formas que se encuentran en el fondo, o a la indumentaria de los sujetos en algunas ocasiones el resultado de la detección facial no es la cara del sujeto tal y como nos la esperábamos.

```
function viola_Visible(total_faces,tx,ty)



% numero total de sujetos--> total_faces

% tx -----> ancho del recuadro a devolver si se usa Viola-Jones

% ty -----> alto del recuadro a devolver si se usa Viola-Jones

% se incrementa total_faces en dos unidades para evitar que saque por

% pantalla las carpetas "." y ".."



directorio_leer = 'D:/Richi/Imagenes Logitech';

directorio_guardar = 'D:/Richi/Matlab/Imagenes Logitech Recortadas';

total_faces = total_faces+2;



%%%%% subdirectorios con imagenes

imag = dir(directorio_leer);



%%%%% (se saltan los dos primeros subdirectorios: '.' y '..')
```

```
for i=3:total_faces


   Figure;



        %%%%% le añado el .jpg

        nombre=sprintf('%s.jpg',imag(i).name);

        %%%%% leemos la imagen

        path = sprintf('%s/%s',directorio_leer,imag(i).name);

        y = imread(path);


        % %%%% pasamos la imagen a niveles de gris

        y2=rgb2gray(y);

        %%%% recortamos con el detector de Viola:

        [imagen_recortada] = recorta_viola(y2, tx, ty, imag(i).name);

        path_guardar= sprintf('%s/%s',directorio_guardar,imag(i).name);

        imwrite(imagen_recortada,path_guardar,'jpeg');


end

fprintf('\n');

return
```

FUNCIÓN RECORTA_VIOLA_FULL

Esta función es el recurso a utilizar si intuimos que la función viola_visible puede tener dificultades para lograr la detección de la cara deseada.

Su objetivo es sacar todos los posibles resultados que, según el criterio del algoritmo de detección Viola-Jones sean un rostro humano.

Generalmente obtendremos no solo el rostro que queremos sino todo el resto de caras que aparezcan así como otro tipo de objetos.

Una vez procesada, guardara las imágenes en el directorio que especifiquemos indicando el nombre de la imagen y el numero de cara detectado. Si procesamos una imagen llamada ejemplo, y detecta cuatro caras están serán etiquetadas como ejemplo_cara1, ejemplo_cara2, ejemplo_cara3, ejemplo_cara4. Tras ello deberemos eliminar manualmente las que no se correspondan al resultado que esperamos y renombrar la correcta.

```
function recorta_viola_full(nombre_imagen, tx, ty)
% nombre_imagen --> nombre de la imagen  sin la extensión (sin el jpg)
%
% tx -----> ancho del recuadro a devolver si se usa Viola-Jones
% ty -----> alto del recuadro a devolver si se usa Viola-Jones

%lectura de la imagen y conversión a niveles de gris
nombre_completo=sprintf('%s.jpg',nombre_imagen);
imagen = rgb2gray(imread(nombre_completo));



% RECORTE DE LA IMAGEN SEGUN VIOLA-JONES
%----------------------------------------

% se llama al programa de deteccion de caras de Viola-Jones y
% se devuelven las esquinas del rectangulo que contiene la imagen

extra = 20; %tamaño del cuadrado extra para que el detector Viola-Jones funcione mejor
[x, y] = size(imagen);
new_im = uint8(zeros(x+2*extra, y+2*extra));
new_im(extra+1:extra+x,extra+1:extra+y) = imagen;

%% esta versión full devuelve un vector de puntos
%% pues detecta todas las caras
[x1 y1 x2 y2 ok] = violajones_full(new_im);
%%%% si no detecta caras, se sale
if ok==0
   disp('Lo siento, pero no veo ninguna cara')
   return;
end
```

```
%numero de caras detectadas
num=length(x1);

% Bucle para todas las caras detectadas
for  i=1:num

   a1=x1(i)+1; %%% corrección del 0
   b1=y1(i)+1;
   a2=x2(i)+1;
   b2=y2(i)+1;

   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   % se recorta el recuadro y se escala a un tamaño fijo
   imagen = new_im(b1:b2,a1:a2);
   imagen = imresize(imagen, [ty tx], 'bilinear');
   imagen_recortada=imagen;

   %%%%% se guardan todas las caras recortadas
   %%%%% dentro de la carpeta "recortadas"
   nombre_cara=sprintf('%s_cara_%d.jpg',nombre_imagen,i);

   cd 'Recortadas full'
   imwrite(imagen_recortada,nombre_cara,'jpeg');
   cd ..

   %%%%% Para visualizarlas dentro de matlab

   see=0;
   if see
      % escalado a 0:255
      imagen = double(imagen);
      low = min(min(imagen));
      high = max(max(imagen));
      im_esc = (imagen-low)*255/(high-low);

      %%%%% dibuja imagen
      Figure; imshow(uint8(im_esc));
   end



end

return
```

FUNCIÓN MONTAJE.

Una vez hemos obtenido todas las fotos de todos los sujetos nos vimos en la necesidad de montar las la imágenes de cada cara de cada sujetos en una sola imagen individual que mostrase todas las caras recortadas de un sujeto.

```
function montaje(ini,fin)


for sub=ini:fin

   %% Inicializamos las matrices que van a contener las imágenes así como
   %% los vectores que definirán el espacio de separación entre imágenes

   matriz=[];
   matriz2=[];
   matriz3=[];
   matriz4=[];
   columna=255*ones(100,10);
   fila=255*ones(10,430);
   separa1=255*ones(100,430);

   %% FOTOS VISIBLES

   for j=1:8

      %% Para el correcto posicionamiento de las imágenes y las
      %% separaciones entre estas dividimos el bucle que se moverá por
      %% las 8 imágenes de cada sujeto en 4 conjuntos.

      if j<4
         comando=sprintf('y=imread(''D:/Richi/Mediterranean
Faces/_VisibleC/s%d_%d.jpg'');',sub,j);

         eval(comando);

         fprintf('s%d_%d.jpeg \n',sub,j)
         matriz=[matriz y columna ];
      end

      if j==4
         comando=sprintf('y=imread(''D:/Richi/Mediterranean
Faces/_VisibleC/s%d_%d.jpg'');',sub,j);
         eval(comando);
         fprintf('s%d_%d.jpeg \n',sub,j)

         matriz=[matriz y;fila];

      end

      if (j>=5 && j<8)
         comando=sprintf('y=imread(''D:/Richi/Mediterranean
Faces/_VisibleC/s%d_%d.jpg'');',sub,j);
         eval(comando);
```

```
        fprintf('s%d_%d.jpeg \n',sub,j)

        matriz2=[matriz2 y columna];
      end
      if j==8
        comando=sprintf('y=imread(''D:/Richi/Mediterranean
Faces/_VisibleC/s%d_%d.jpg'');',sub,j);
        eval(comando);
        fprintf('s%d_%d.jpeg \n',sub,j)

        matriz2=[matriz2 y];
      end



  end

  matriz=[matriz;matriz2;separa1];


%% FOTOS WILD LABORATORIO
%% Al igual que en el caso de las 8 visibles para las cuatro
%% tomadas con iPhone y las cuatro libres.

  for j=1:8

    if j<4
      comando=sprintf('y=imread(''D:/Richi/Mediterranean Faces/_WildC/s%d_%d.jpg'');',sub,j);
      eval(comando);
      fprintf('s%d_%d.jpeg \n',sub,j)
      matriz3=[matriz3 y columna ];
    end
    if j==4
      comando=sprintf('y=imread(''D:/Richi/Mediterranean Faces/_WildC/s%d_%d.jpg'');',sub,j);
      eval(comando);
      fprintf('s%d_%d.jpeg \n',sub,j)

      matriz3=[matriz3 y];
    end

%% FOTOS WILD LIBRES

    if (j>=5 && j<8)
      comando=sprintf('y=imread(''D:/Richi/Mediterranean Faces/_WildC/s%d_%d.jpg'');',sub,j);
      eval(comando);
      fprintf('s%d_%d.jpeg \n',sub,j)

      matriz4=[matriz4 y columna];
    end
    if j==8
      comando=sprintf('y=imread(''D:/Richi/Mediterranean Faces/_WildC/s%d_%d.jpg'');',sub,j);
      eval(comando);
      fprintf('s%d_%d.jpeg \n',sub,j)

      matriz4=[matriz4 y];
    end



  end
```

```
   %% Montamos una matriz que contenga todas las demás así como
   %% los espacios que separan las matrices con distintos tipos de
   %% imágenes.

   matriz=[matriz;matriz3;separa1;matriz4];
   Figure(sub)

   %% Nombramos la imagen como "s+numerosujeto"

   titulo=sprintf('s%d.jpg',sub);

   %% Por último guardamos la imagen con el titulo que acabamos de
   %% establecer en formato JPEG.

   imwrite (matriz,titulo,'jpeg');
end
```

FUNCIÓN MAIN_CROPPEAR.

Tuvimos acceso a esta función una vez finalizado el preprocesamiento de todas las imágenes. Debido a su fácil y ágil manejo esta función , que contiene el algoritmo de detección Viola-Jones, es de enorme utilidad para realizar el recorte de caras selectivo.

Una vez ejecutada, la función saca por pantalla todas las caras de ocho imágenes del directorio que hayamos indicado. Simplemente debemos seleccionar con el cursor cual de las caras es la que se corresponde al sujeto que queremos en nuestra base de datos.

Además de realizar el recorte con las medidas que hayamos especificado, guarda la posición de la cara en forma de dos pares de coordenadas, la de la esquina superior izquierda y la de la inferior derecha. Estas coordenadas no tienen en cuenta el tamaño del recorte, si no que son el producto directo de la detección sobre la imagen original.

La utilidad de estas coordenadas es principalmente la de ahorrar tiempo de procesamiento, ya que, este mismo programa, en caso de que volvamos a realizar la detección y localice la existencia del archivo de coordenadas, acudiría a la posición indicada por estas, evitando así la aplicación del algoritmo.

```
% Recorre un directorio de fotos originales y recorta el área de la cara
%
% Guarda las imágenes recortadas en otro directorio
% imágenes y se muestran los resultados en pantalla para verificar que son
% correctos. Luego se guarda la imagen recortada en el nuevo directorio y
% las coordenadas del recorte en un fichero de texto.
%
% Si se detecta que ya se ha ejecutado Viola-Jones (porque existe el
% fichero de coordenadas), se recorta y se almacena la imagen recortada sin
% ejecutar Viola-Jones de nuevo y sin pedir confirmación.

% parámetros:
% - dir_ori: directorio con las imágenes originales
% - dir_dest: directorio donde guardar las imágenes recortadas
% - tam: pixels de la imagen recortada: ej. poner tam=100 para 100x100

function main_croppear(dir_ori, dir_dest, tam)

% buscamos las imágenes en el directorio origen
buscar = sprintf('%s/*.jpg', dir_ori);
lista = dir(buscar);
num_im = length(lista);

% inicializaciones:
proc = 0; % contador de imágenes procesadas

% bucle para cada imagen
for i=1:num_im

  % leemos la imagen
```

```
  fprintf('Procesando imagen %s...\n', lista(i).name);
  path_im = sprintf('%s/%s', dir_ori, lista(i).name);
  imagen = imread(path_im);
  % size(imagen)

  % nombre del fichero de coordenadas
  fichero_coord = sprintf('%s/coord_%s', dir_ori, lista(i).name);
  fichero_coord = strrep(fichero_coord, '.jpg', '.txt');
  fichero_coord = strrep(fichero_coord, '.JPG', '.txt');

  % comprobamos si existe y tiene datos correctos
  correcto = 0;
  file = fopen(fichero_coord, 'r');
  if file ~= -1
    linea = fgets(file);
    coordenadas = sscanf(linea, '%d');
    if length(coordenadas)==4
      correcto = 1;
    end
    fclose(file);
  end

%%%%  si existe el fichero, utilizamos las coordenadas para recortar
  if correcto

    % coordenadas de la cara
    x_ini = coordenadas(1);
    y_ini = coordenadas(2);
    x_fin = coordenadas(3);
    y_fin = coordenadas(4);

    % recorte y escalado
    recorte = imagen([y_ini:y_fin], [x_ini:x_fin], :);
    recorte = imresize(recorte, [tam, tam], 'bilinear');

    % guardamos la imagen
    fichero_destino = sprintf('%s/%s', dir_dest, lista(i).name);
    imwrite(recorte, fichero_destino);

% si no existe el fichero, llamamos a Viola-Jones
  else

    extra = 20; %tamaño del cuadrado extra para que el detector

    % Viola-Jones funcione mejor

    [x, y, z] = size(imagen);
    new_im = uint8(zeros(x+2*extra, y+2*extra,z));
    new_im(extra+1:extra+x,extra+1:extra+y,:) = imagen;

    %% esta versión full devuelve un vector de puntos
    %% pues detecta todas las caras
    [x1 y1 x2 y2 ok] = violajones_full(new_im);

    % si no detecta caras, pasamos a la siguiente imagen
    if ok==0
      fprintf('NO DETECTADA CARA!!!!!!!!!!!')
      continue
    end

    % guardamos los datos de la detección
```

```
proc = proc+1;
vx1{proc} = [x1];
vx2{proc} = [x2];
vy1{proc} = [y1];
vy2{proc} = [y2];
indice_im(proc) = i;

% si se han procesado ya 8 imágenes (o si es la última imagen)
% se muestra un plot para que elijamos los recortes correctos

if proc==8 || i==num_im

   % calculamos el número de columnas del subplot
   for j=1:proc
      num_caras(j) = length(vx1{j});
   end
   ancho = max(num_caras);

   % creamos la superimagen en negro
   espacio = 20;
   alto_im = tam*proc + espacio*(proc+1);
   ancho_im = tam*ancho + espacio*(ancho+1);
   superimagen([1:alto_im],[1:ancho_im],[1:3]) = 0;
   superimagen = uint8(superimagen);

   % mostramos todas las imágenes
   for j=1:proc
      path_im = sprintf('%s/%s', dir_ori, lista(indice_im(j)).name);
      imagen = imread(path_im);
      [im_y, im_x, im_z] = size(imagen);
      for k=1:length(vx1{j})

         % restamos el recuadro externo

         x1ok = max(1, vx1{j}(k)-extra);
         y1ok = max(1, vy1{j}(k)-extra);
         x2ok = min(im_x, vx2{j}(k)-extra);
         y2ok = min(im_y, vy2{j}(k)-extra);
         recorte = imagen([y1ok:y2ok], [x1ok:x2ok], :);
         recorte = imresize(recorte, [tam tam], 'bilinear');

         % añadimos el recorte a la superimagen

         alto_ini = espacio*j + tam*(j-1);
         alto_fin = alto_ini + tam - 1;
         ancho_ini = espacio*k + tam*(k-1);
         ancho_fin = ancho_ini + tam - 1;
         superimagen([alto_ini:alto_fin], [ancho_ini:ancho_fin], :) = recorte(:, :, :);

      end
   end

   Figure(1);
   imshow(superimagen);

   % dejamos que el usuario pinche sobre las correctas
   cadena = sprintf('íPINCHE SOBRE LAS %d IM⊢GENES CORRECTAS!', proc);
   fprintf('%s\n', cadena);
   title(cadena);
   pinchazos = ginput(proc);
   close(1);
```

```
        clear superimagen;

        % evaluamos sobre que imagen ha pinchado en cada caso
        for j=1:proc
          elegido(j) = ceil(pinchazos(j,1)/(tam+espacio));
          fprintf(' - Fila %d, elegida imagen %d\n', j, elegido(j));
        end

        % guardamos las imágenes recortadas adecuadas y las coordenadas
        for j=1:proc

          % primero, la imagen

          path_im = sprintf('%s/%s', dir_ori, lista(indice_im(j)).name);
          imagen = imread(path_im);
          [im_y, im_x, im_z] = size(imagen);

          % restamos el recuadro externo

          x1ok = max(1, vx1{j}(elegido(j))-extra);
          y1ok = max(1, vy1{j}(elegido(j))-extra);
          x2ok = min(im_x, vx2{j}(elegido(j))-extra);
          y2ok = min(im_y, vy2{j}(elegido(j))-extra);

          recorte = imagen([y1ok:y2ok], [x1ok:x2ok], :);
          recorte = imresize(recorte, [tam tam], 'bilinear');

          fichero_destino = sprintf('%s/%s', dir_dest, lista(indice_im(j)).name);
          imwrite(recorte, fichero_destino);

          % ahora, las coordenadas

          fichero_coord = sprintf('%s/coord_%s', dir_ori, lista(indice_im(j)).name);
          fichero_coord = strrep(fichero_coord, '.jpg', '.txt');
          fichero_coord = strrep(fichero_coord, '.JPG', '.txt');
          file = fopen(fichero_coord, 'w');
          fprintf(file, '%d %d %d %d\n', x1ok, y1ok, x2ok, y2ok);
          fclose(file);

        end

        proc=0; % para poder empezar a contar otra vez
      end

    end

end


            return
```

4.3. ACONDICIONAMIENTO DEL LABORATORIO

La toma de las fotos posadas, tanto visibles como infrarrojas, ha sido realizada en el laboratorio de control de sistema inteligentes del edificio Quórum V del campus de Elche de la Universidad Miguel Hernández de Elche. De esta manera hemos podido realizar la captura en un ambiente controlado.

Para la captura de las imágenes infrarrojas nos encontramos ante el contratiempo de no disponer de una sala oscura donde poder poner en funcionamiento la opción infrarroja de la cámara de vigilancia.

La solución adoptada fue la de acondicionar una zona del laboratorio impidiendo la entrada de luz a dicha zona tal y como se muestra en la imagen.

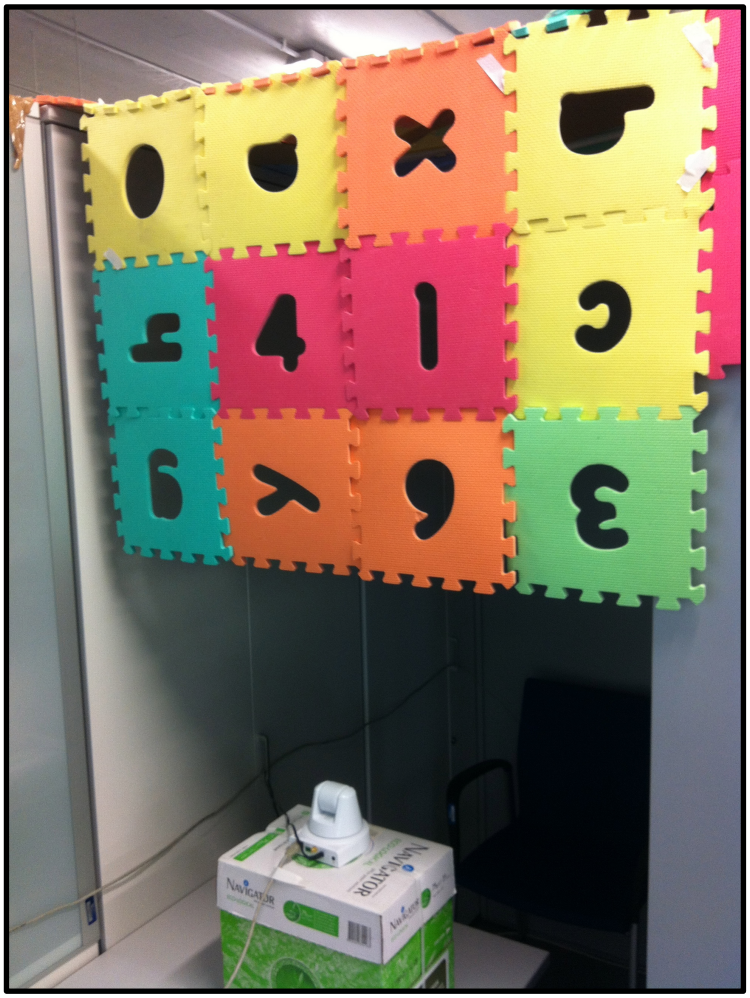

Figura 4.9 – Montaje para las capturas en IR -

MEDIDAS PARA LA TOMA DE IMÁGENES.

Con el fin de buscar la máxima homogeneidad entre las fotos del mismo tipo se fijaron las distancias y posiciones que iba a ocupar cada elemento a la hora de la toma de las fotos posadas.

Figura 4.10 – Momento de captura de fotos tipo Visible -

Para ello primero se marco la posición de la mesa donde se depositaria la cámara tal y como podemos observar en la Figura 4.4.

Figura 4.11 – Marca para fijación de la cámara Logitech -

El objetivo de la cámara se encuentra a una altura de 1 metro sobre el suelo.

Figura 4.12 – Medida Objetivo-Suelo -

A partir de ahí y a 80 cm de distancia se establece la posición de la silla donde se situarán los sujetos.

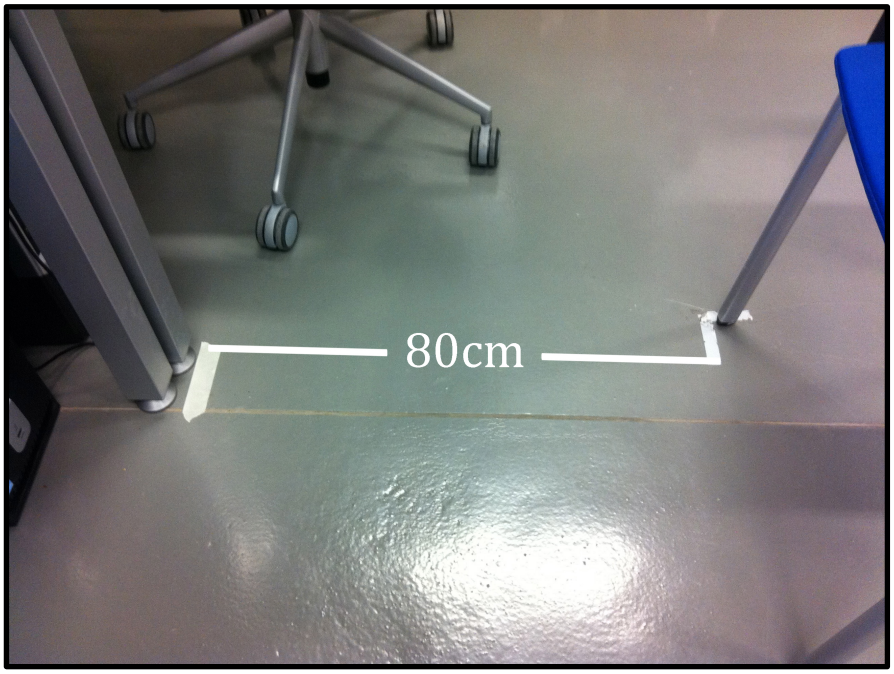

Figura 4.13 – Medidas Mesa-Objetivo -

En el caso de las imágenes no posadas, y dada la naturaleza espontanea de estas fotos se realizaron en diferentes zonas tanto del laboratorio como del resto del edificio, buscando de este modo la máxima variedad de fondos e iluminaciones.

4.4. CARACTERÍSTICAS Y PUESTA EN FUNCIONAMIENTO DE LAS CÁMARAS.

4.4.1. IMÁGENES VISIBLES

La cámara de luz visible con la que se tomaran las imágenes es el modelo Sphere AF de Logitech. Posee una resolución de 2Mpx y un abarca un ángulo de captura de 170º en el plano horizontal y 70 en el plano vertical. Ofrece la posibilidad de acoplarle un adaptador a modo de cuello de unos 30 cm. El software por defecto es compatible con Windows Xp y Vista, y desde el sitio web del fabricante podemos descargar el compatible con Windows 7.

Una vez ejecutamos el software de la cámara se abre la siguiente interfaz:

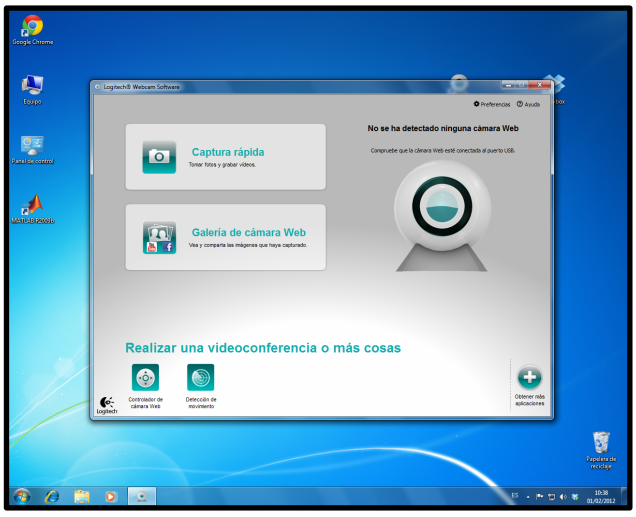

Figura 4.14 – Captura de pantalla inicial del driver de la cámara Logitech Sphere -

Desde la pantalla de inicio podemos acceder al menú de captura rápida, Galería de imágenes , preferencias , detección de movimiento y al menú de control.

Dentro del menú de captura rápida podemos cambiar de modo video a modo foto , añadir efectos y acceder directamente al menú de control.

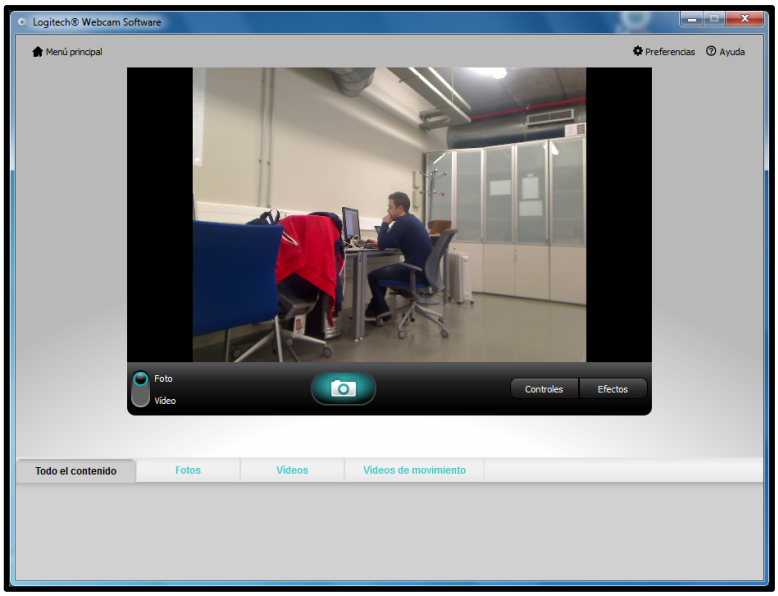

Figura 4.15 – Interfaz para la toma de fotos manual -

Para detección de movimiento se debe fijar el umbral de sensibilidad donde se estime oportuno. Así como la duración de la grabación cuando la detección se haya llevado a cabo. Figura X.

Para el control remoto las opciones son giro derecha, giro izquierda, giro arriba, giro abajo, acercar y alejar. Desde este menú también podremos modificar la resolución de la cámara entre grande (800x600), media (640x480), pequeña (320x240), así como seleccionar las opciones de enfoque automático. Figura X.

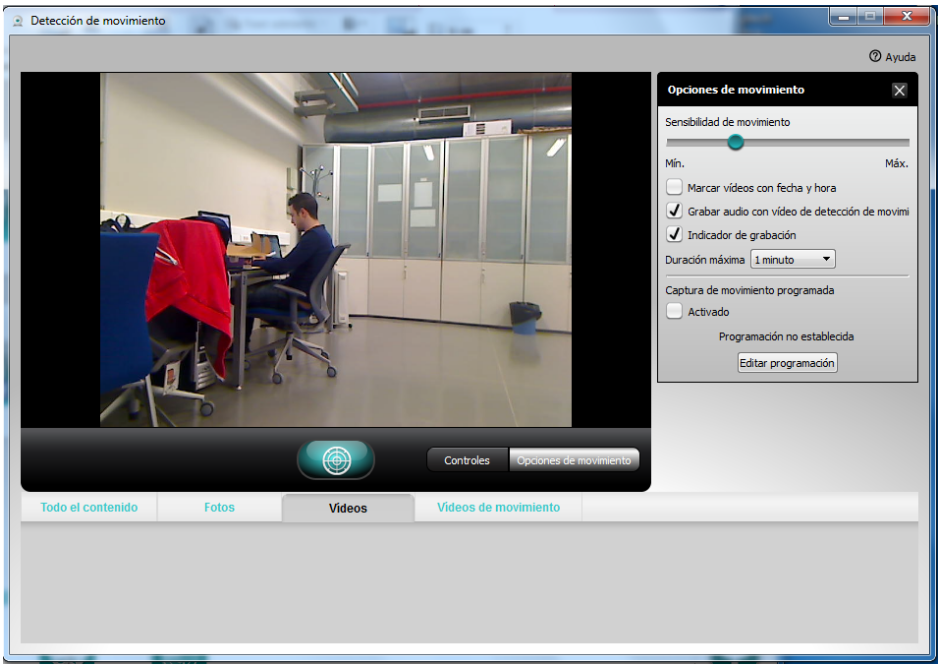

Figura 4.16 – Pantalla del sistema de detección de movimiento -

Figura 4.17 – Pantalla de control remoto de la cámara Logitech Sphere -

Dentro de configuración avanzada se modifican la exposición, la ganancia, brillo, contraste, intensidad de color, equilibrio de blanco, orientación normal o reflejada y control de parpadeo (Figura X). Se dejaran fijados en los valores por defecto

Figura 4.18 – Pantalla de configuración -

En el menú de preferencias se puede seleccionar la carpeta de destino de las imágenes y videos tomados, así como modificar las calidades de grabación de video, imagen y audio.

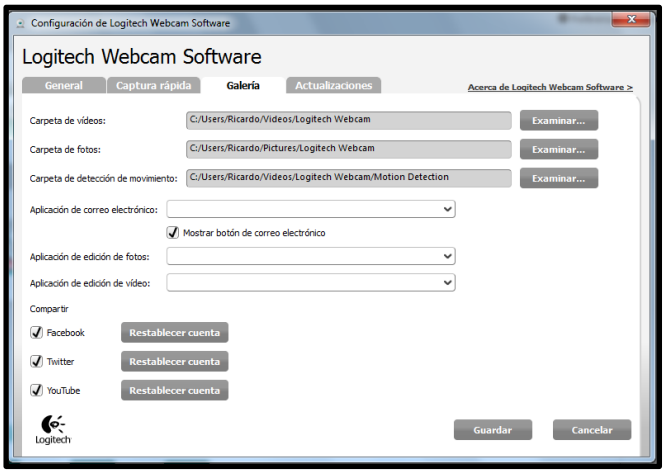

Figura 4.19 – Pantalla de configuración de directorios -

4.4.2. IMÁGENES WILD

Las imágenes no posadas o *Wild* tomadas en el laboratorio se capturaran con la cámara de un dispositivo móvil IPhone 4 [13].

Las características de esta cámara son:

- Grabación de video en HD (720p) hasta a 30 cuadros por segundo, con audio.
- Cámara de 5 megapíxeles
- Resolución VGA de fotos y video a 30fps con la cámara frontal
- Tocar para enfocar en fotos y video
- LED flash
- Geotagging de foto y video.

Para el traspaso de las imágenes desde el dispositivo móvil iPhone 4 al ordenador se ha utilizado el software iCloud. Esta tecnología nos permite recibir las imágenes en un directorio previamente seleccionado y de esta manera no tener que traspasar las imágenes cada vez que deban ser procesadas.

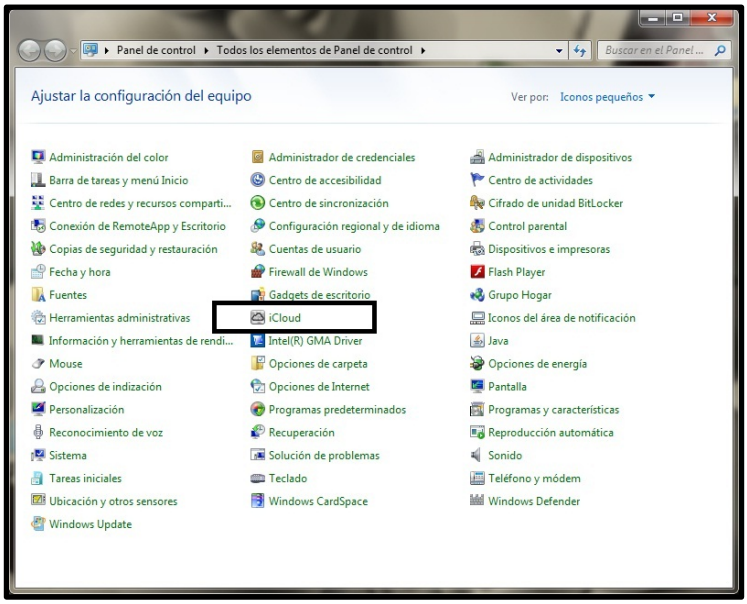

Figura 4.20 –Captura de pantalla del panel de control de Windows 7 con la opción iCloud -

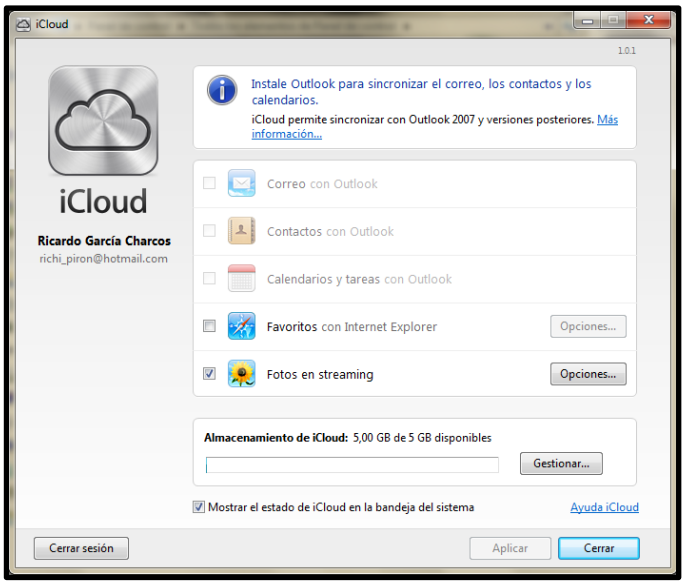

Figura 4.21 –Panel de control iCloud-

4.5. CARACTERÍSTICAS DE LAS IMÁGENES.

En este apartado analizaremos las características de cada clase de imagen.

En el caso de las imágenes posadas, tanto *Visible* como *IR* serán propiedades muy homogéneas, con pequeñas variaciones de tamaño. Mientras que en el caso de las fotos no posadas o *Wild* encontramos variaciones muy significativas en tamaño u dimensiones.

- Imágenes tipo *Visible:*
    - Formato: JPEG
    - Color: RGB
    - Dimensión: 1600x1200
    - Tamaño: 344Kb – 573Kb

- Imágenes tipo *Wild:*
    - Formato: JPEG
    - Color: RGB

Subdividimos las imágenes del tipo Wild en dos categorías:

- IPhone:
    - Dimensión: 1936x2592 (2592x1936)
    - Tamaño: 696Kb-2,3Mb

- Libres:
    - Dimensión: -
    - Tamaño: 25Kb-5,5Mb

- Imágenes tipo VisibleC (Recortadas):
    - Formato: JPEG

- Color: RGB
- Dimensión: 100x100
- Tamaño: 344Kb – 573Kb

- Imágenes tipo WildC (Recortadas):

  - Formato: JPEG
  - Color: RGB

- IPhone:

  - Dimensión: 100x100
  - Tamaño: 2Kb

- Libres:

  - Dimensión: 100x100
  - Tamaño: 2Kb

4.6. VENTAJAS Y CARACTERÍSTICAS DE MEDITERRANEAN FACES.

Como ya hemos mencionado antes el principal objetivo de la base de datos Mediterranean Faces es el de proporcionar un material de calidad orientado sobre todo al reconocimiento facial y a la detección facial. Para ello nos fijamos como objetivo que, de cada sujeto, pudiésemos tener varios tipos de imagen pudiendo así realizar combinaciones a la hora de entrenar el algoritmo. La utilidad de esto puede no quedar clara en primera instancia pero si analizamos la situación real en el campo de la vigilancia todo cobra sentido. Supongamos que queremos localizar una persona buscada por las autoridades, difícilmente tendremos en el archivo imágenes de la susodicha persona en infrarrojo. Con Mediterranean Faces tenemos la posibilidad de realizar comprobaciones entrenando con las imágenes de cámara visible y realizando los test con las imágenes infrarrojas.

La principal ventaja de la base de datos Mediterranean Faces es su gran versatilidad; encontramos fotos de distintos tamaños, resoluciones y sobre todo de tipos de imágenes.

En el caso de la mayoría de bases de datos las imágenes son frontales, con pocas variaciones o en el mejor de los casos variaciones en la iluminación. Obviamente el resultado será de un alto porcentaje de detección/reconocimiento. No obstante, en la practica real del reconocimiento facial, sobretodo para el campo de la vigilancia, no solemos encontrar tomas totalmente frontales del rostro de los sujetos.

En otras bases de datos, como sería el caso de la base de datos Chokepont, encontramos imágenes de distintos ángulos, que en principio solucionaría esta carencia, pero tampoco cubriría las necesidades que intentamos dar con nuestra base de datos, ya que, en el entorno de la detección facial , el algoritmo de Viola-Jones no detecta como cara los rostros que estén girados a partir de aproximadamente 20º en referencia al objetivo.

Gracias a la gran variedad de imágenes, podremos tanto hacer comprobaciones con imágenes de escasa dificultad con las posadas frontales, tanto infrarrojas como con luz visible, o poner a prueba la viabilidad de algoritmos cuyo nivel de exigencia sea superior con las fotos no posadas o *Wild.*

Otra de las ventajas viene dada por la variedad en las resoluciones de las imágenes. El motivo de esto es que, para temas de vigilancia, normalmente las imágenes serán en movimiento, o directamente capturas

de videos por lo que la calidad de la imagen se vera seriamente afectada por este hecho.

La única base de datos que cubre parte de estas características que hemos planteado como necesidades en nuestro caso es *Labeled Faces in the Wild.* Esta base de datos de diferencia de las demás precisamente en que no se basa en imágenes de sujetos posando frente a la cámara, si no que son imágenes seleccionadas de personajes públicos en diversas situaciones, con distintos atuendos y en distintos escenarios. Como factor añadido la calidad de las fotos varia, y muchas son imágenes tomadas cuando el sujeto esta en movimiento.

Del estudio de esta base de datos surgió la idea de incorporar en una única librería este tipo de imágenes así como las posadas.

El caso análogo en Mediterranean Faces vendría dado por las imágenes tipo *Wild* con subíndices del 5 al 8, es decir, para el sujeto 12 serian *s12_5, s12_6, s12_7, s12_8.*

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

El apartado *Wild* es completado por imágenes no posadas y capturadas en el laboratorio con la cámara de un dispositivo móvil IPhone 4. De esta manera las imágenes serán en posiciones más naturales, y espontaneas, permitiendo así el estudio de fotos de características similares a las Wild pero con un formato similar, una resolución superior, y en la mayoría de los casos un ambiente contralado. Aunando así ventajas tanto de las imágenes del tipo de la librería *Labeled Faces in the wild* como de las posadas que encontramos en las librerías convencionales. Estas imágenes vienen etiquetadas dentro de la carpeta *Wild* con los subíndices del 1 al 2. Siendo, por ejemplo, para el sujeto 12: *s12_1, s12_2, s12_3, s12_4.*

Figura 4.22 –Imágenes espontaneas - IPhone del sujeto 12-

Por otra parte, en cuanto a las imágenes con luz visible, es decir, el tipo *Visible,* hemos seguido un criterio distinto. Buscando la diferenciación de las demás bases de datos mediante la inclusión diferencias gestuales entre las imágenes de un mismo sujeto. Este tipo de imágenes serviría para software de reconocimiento de sonrisas o de otra de los 4 gestos distintos; siendo estos:

1. Neutro.

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

2. Sonrisa

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

3. Sorpresa

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

4. Tristeza

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

5. Enfado.

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

En el caso del gesto *neutro* o normal, se toma por duplicado, ya que, hemos dado por hecho que seria el más frecuente. Y siendo la segunda la sexta imagen de cada sujeto.

6. Neutro 2.

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

Por otra parte, se añaden 2 imágenes más, nombradas *ligeras oclusiones* , estas imágenes añaden complementos o accesorios como gafas, gorras, o bufandas, para dificultar el reconocimiento.

7. Ligeras oclusiones .

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

8. Ligeras oclusiones 2.

Figura 4.22 –Imágenes espontaneas - libres del sujeto 12-

En caso del resto de librerías encontramos librerías de expresión monótona y en muchos casos tomadas con diferencias de tiempo significativamente cortas, siendo estas por lo tanto prácticamente iguales y facilitando el reconocimiento o la detección pero otra parte los resultados serán menos relevantes.

En algunas encontramos variaciones de iluminación de todo tipo, aportando así dificultad a los algoritmos que deseen identificar esa cara o detectarla. Como ya hemos visto en el punto 3 este es el caso de las librerías PIE, AR Faces y la base de datos de la universidad de Yale.

Otra ventaja, tal vez menos significativa sea que el rango de edad es algo más amplio que en otras librerías. Estas suelen abastecerse de imágenes del profesorado del instituto técnico o universidad correspondiente, mientras que en el caso de Mediterranean Faces hemos incluido imágenes de sujetos más jóvenes, siendo estas complementadas por otros sujetos de mayor edad. Siendo la media de edad de 25,54 , el sujeto de mayor edad 55 y el de menor 19.

En cuanto al sexo de los sujetos hemos logrado alcanzar prácticamente la paridad, siendo la proporción de 58% hombres y 42% mujeres. Pese a la aparente nimiedad de este hecho, es la base de datos que más se acerca a la paridad entre hombres y mujeres.

En ciertos apartados no cubrimos las necesidades que otras si cubren, como es el caso de la variedad étnica, las variaciones controladas de iluminación o de ángulo. En caso de la variedad étnica el motivo ha sido obviamente la escasa variedad de sujetos a la que hemos podido tener acceso. Mientras que en la variación de ángulo y de iluminación ha sido por falta de medios.

5. PROCESADO DE IMÁGENES

5.1. ALGORITMO VIOLA-JONES

Como ya hemos ido mencionado a lo largo de la memoria, el algoritmo de detección Viola-Jones fue creado con el objetivo de proporcionar tasas de detección aceptables por Paul viola y Michael Jones [14]. Aunque la motivación principal que llevo a la creación de este algoritmo fue la de resolver el problema de la detección facial, con el entrenamiento adecuado puede ser utilizado como un algoritmo de detección de otro tipo de objetos.

Este algoritmo es un método robusto y rápido basado en dos fases, el entrenamiento y la detección.

Para el entrenamiento se crean los llamados clasificadores fuertes o *strog classifiers* , estos elementos son los encargados de decidir que partes de la imagen corresponden a una cara y cuales no. Estos clasificadores se organizan en una estructura de cascada, aumentando asi la eficiencia del algoritmo y con ello la velocidad de procesamiento, ya que , gracias a este sistema los objetos son cribados antes cuanto menos sea su semejanza a una cara, eliminando asi la mayoría de objetos en los primeros niveles de la cascada [15] [16].

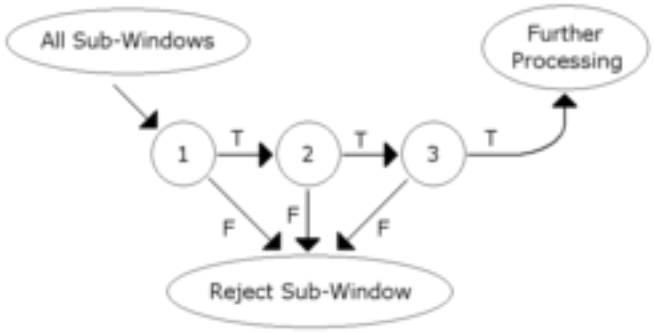

Figura 5.1 – Esquema de estructura de cascada -

Para este algoritmo se emplean características similares a las del ámbito de la detección de objetos. Estas características llamadas características simples o *simple features* se utilizan dentro de lo que se conoce como imagen integral .

Se busca determinar características basadas en sumas y restas de los niveles de intensidad de la imagen, el valor de una característica esta dado por la diferencia de todos los pixeles de color negro con la suma de todos los pixeles de color blanco.

El valor de la imagen integral en la posición *x,y* contiene la suma de todos los pixeles arriba y a la izquierda de *x,y,* ambos inclusive .

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y')$$

$$ii(x,y) = i(x,y) - ii(x-1,y) - ii(x,y-1) + ii(x-1,y-1)$$

Usando la imagen integral cualquier suma rectangular puede ser calculada con cuatro referencias a memoria.

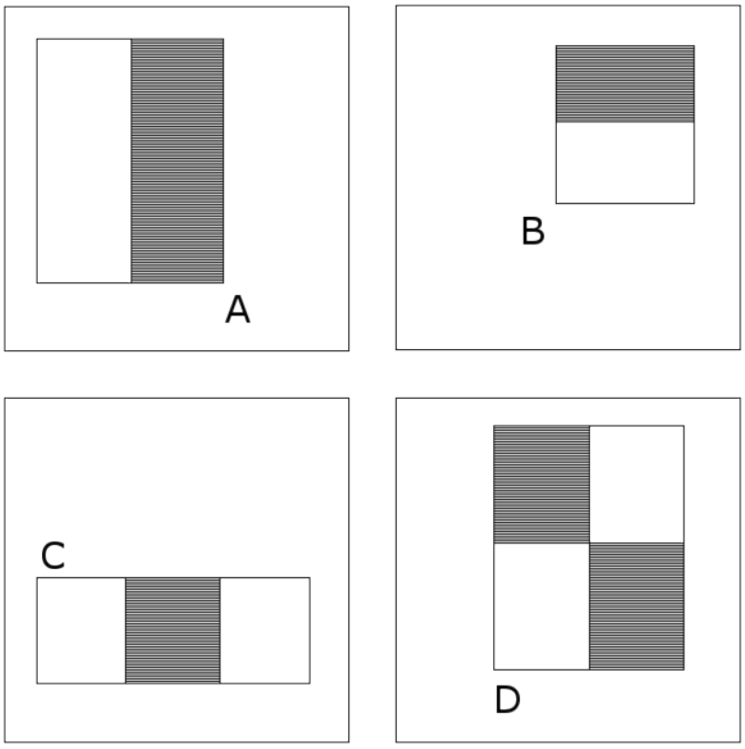

Figura 5.2 –Ejemplo de características simples o simple features -

La eficiencia computacional de las características compensa ampliamente su limitada flexibilidad. El conjunto de estas características tienen la propiedad de que pueden ser evaluadas a cualquier escala con unas cuantas operaciones.

Cada una de las características puede ser calculara de manera eficiente, pero el calculo del conjunto completo es bastante costoso. Por ello se trabaja con un numero reducido, partiendo de la hipótesis de que un numero limitado de características puede ser combinado para formar un clasificador eficiente. Es decir, con una selección de funciones de clasificación débiles formamos un clasificador fuerte.

Finalmente el clasificador fuerte es una combinación de clasificadores débiles y un umbral, en el que, según la experiencia del entrenamiento se asigna mayor peso a los clasificadores que tengan mas éxito y menos peso al resto.

5.2. APLICACIÓN DEL ALGORITMO VIOLA-JONES

Durante el resto de la memoria y especialmente durante el apartado anterior hemos nombrado y analizado el funcionamiento del algoritmo Viola-Jones. En este apartado mostraremos la interfaz que hemos utilizado para así facilitar el etiquetado de las imágenes una vez las procesamos manualmente.

El primer paso tras la toma de imágenes es comprobar que en las imágenes tomadas se produzca la detección positiva de la cara del sujeto. En algunos de los casos el algoritmo devuelve lo que se conoce como falsos positivos, es decir, reconoce como cara objetos que en realidad no lo son. Y en otros, donde encontramos más de un sujeto no tiene porque devolver la cara deseada.

Este primer paso se realiza utilizando la función *viola_visible.* A continuación, En los casos en los que la detección de las imágenes tipo visible no ha sido correcta, y en la totalidad de las imágenes no posadas, utilizamos la función *recorta_viola_full.*

Esta función nos devuelve las imágenes de todos los positivos que se encuentren en la foto, entre los que en la mayoría de los casos se encuentra el que buscamos. La pega de esta función es que debemos eliminar todas las imágenes falsas y renombrar la que queremos conservar.

Como solución a este inconveniente conseguimos utilizar una función cuyo funcionamiento es similar al de *recorta_viola_full* pero con una interfaz mejorada, *main_croppear.*

Esta función saca por pantalla todos los positivos de cada imagen y únicamente debemos seleccionar con el cursor los que sean correspondientes al sujeto que buscamos.

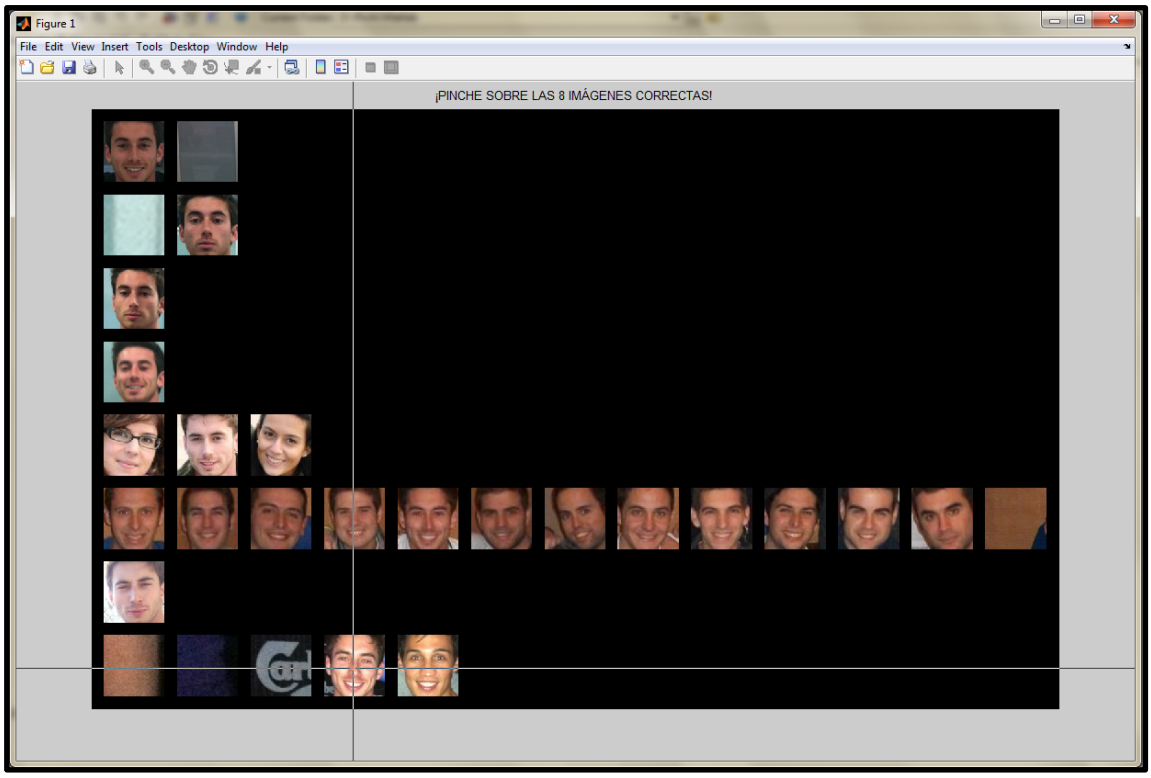

Figura 5.3–Captura de pantalla de la interfaz de la función main_croppear –

Otra de las novedades de esta función es que, como ya vimos en el apartado 4.1 la función que contiene el algoritmo Viola-Jones nos permite aprovechar las coordenadas que esta genera ahorrando así tiempo de procesamiento.

Para el manejo de esta función simplemente debemos indicar el directorio de origen y de destino así como el tamaño del recuadro que queremos recortar.

6. LIBRERÍA IMÁGENES

Fig. 6.1 – Conjuntos de imágenes sujeto 1

Grupo A: Imágenes tipo Visible

Grupo B: Imágenes tipo Wild-IPhone

Grupo C: Imágenes tipo Wild-Libre

Fig. 6.2 – Conjuntos de imágenes sujeto 2 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

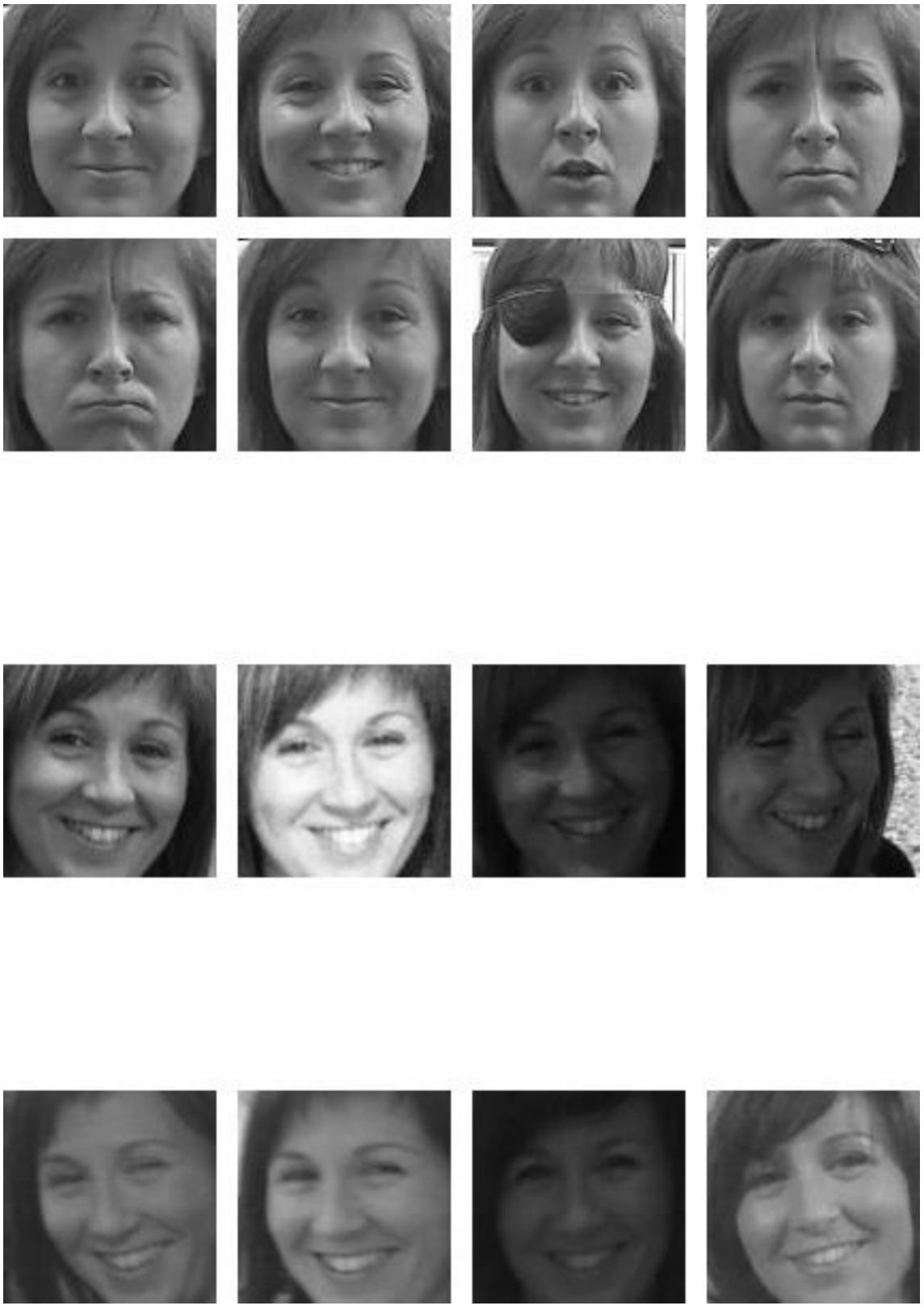

Fig. 6.3 – Conjuntos de imágenes sujeto 3 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.4 – Conjuntos de imágenes sujeto 4 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.5– Conjuntos de imágenes sujeto 5 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

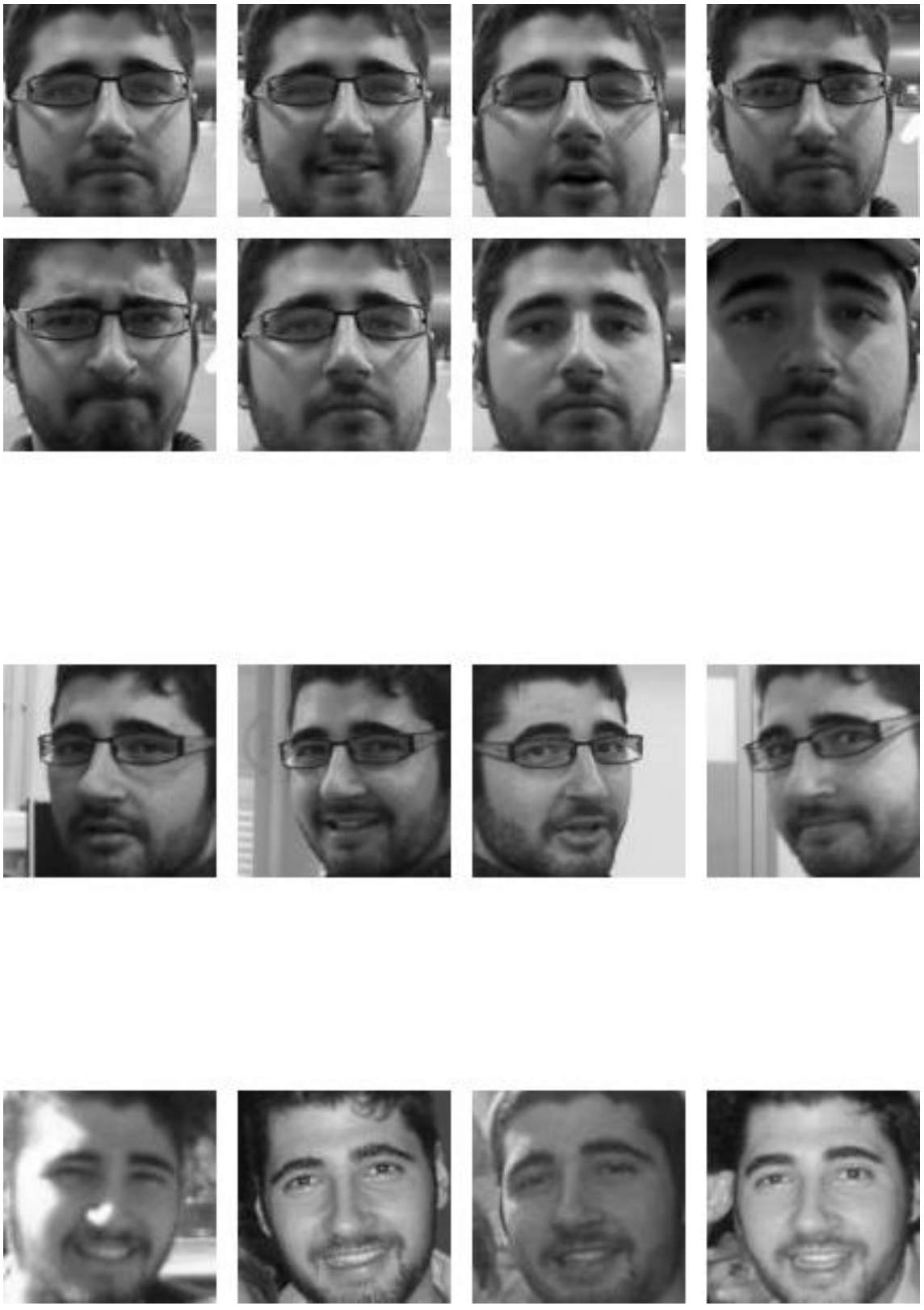

Fig. 6.6 – Conjuntos de imágenes sujeto 6 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

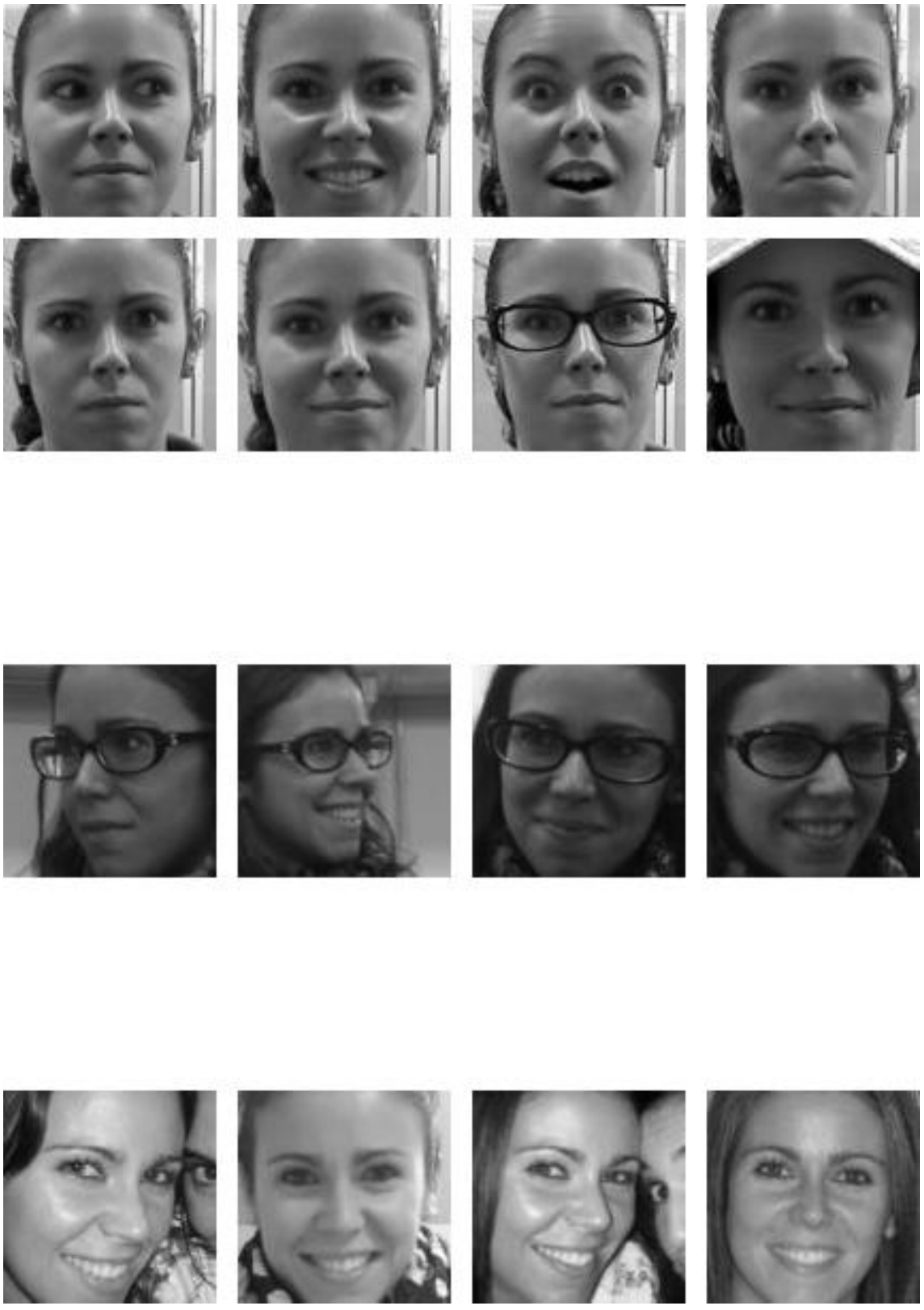

Fig. 6.7 – Conjuntos de imágenes sujeto 7 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

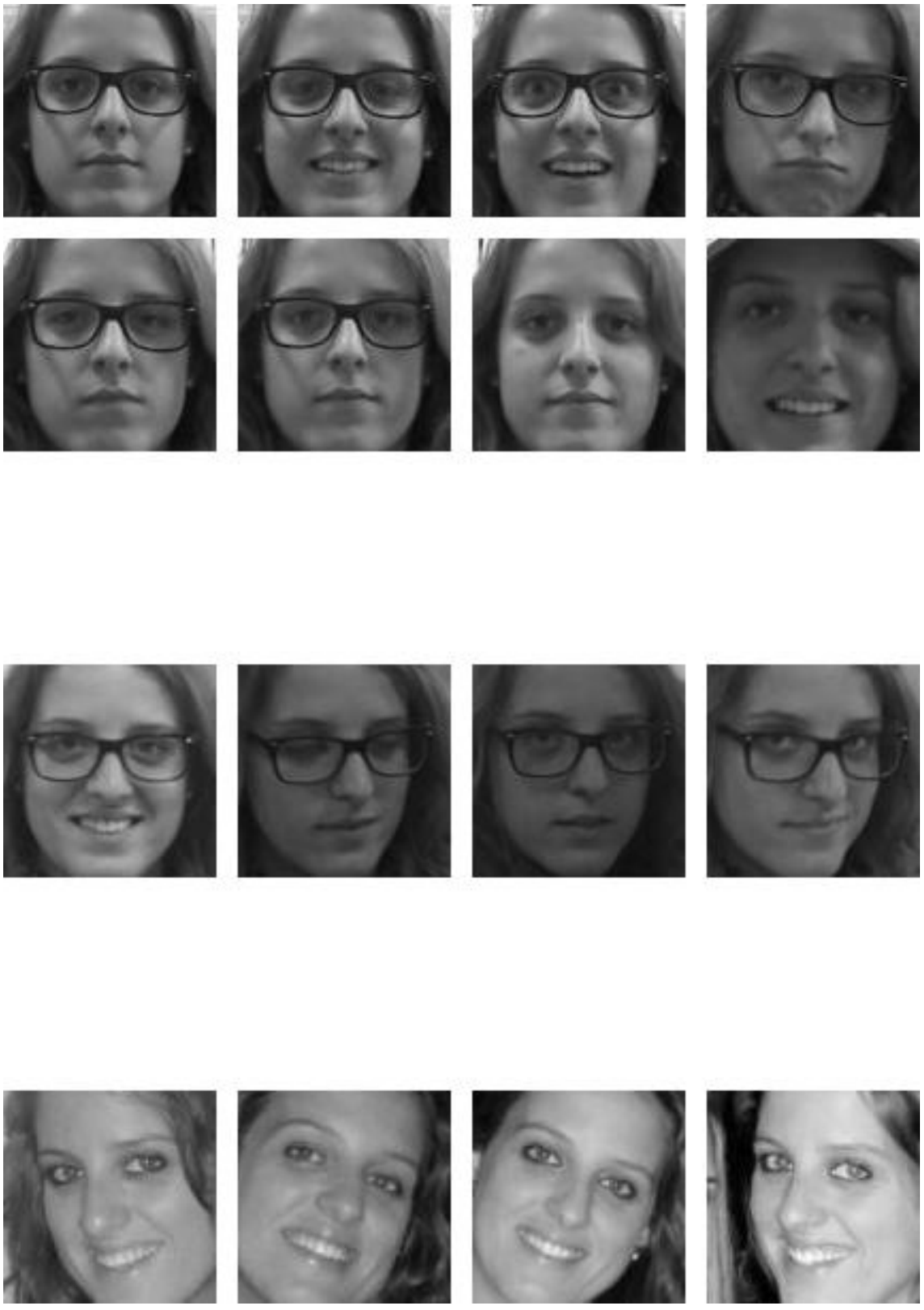

Fig. 6.8 – Conjuntos de imágenes sujeto 8 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

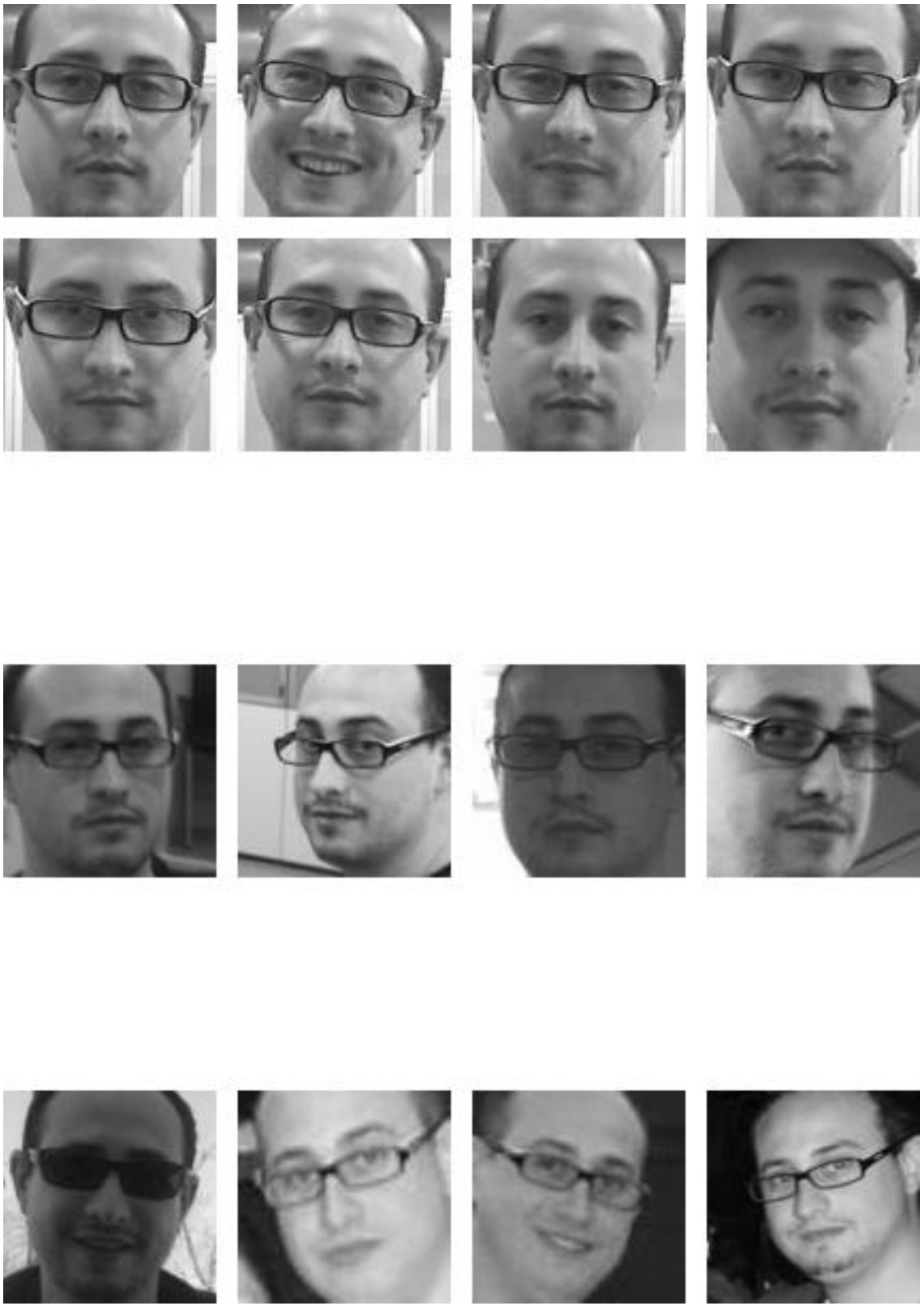

Fig. 6.9 – Conjuntos de imágenes sujeto 9 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

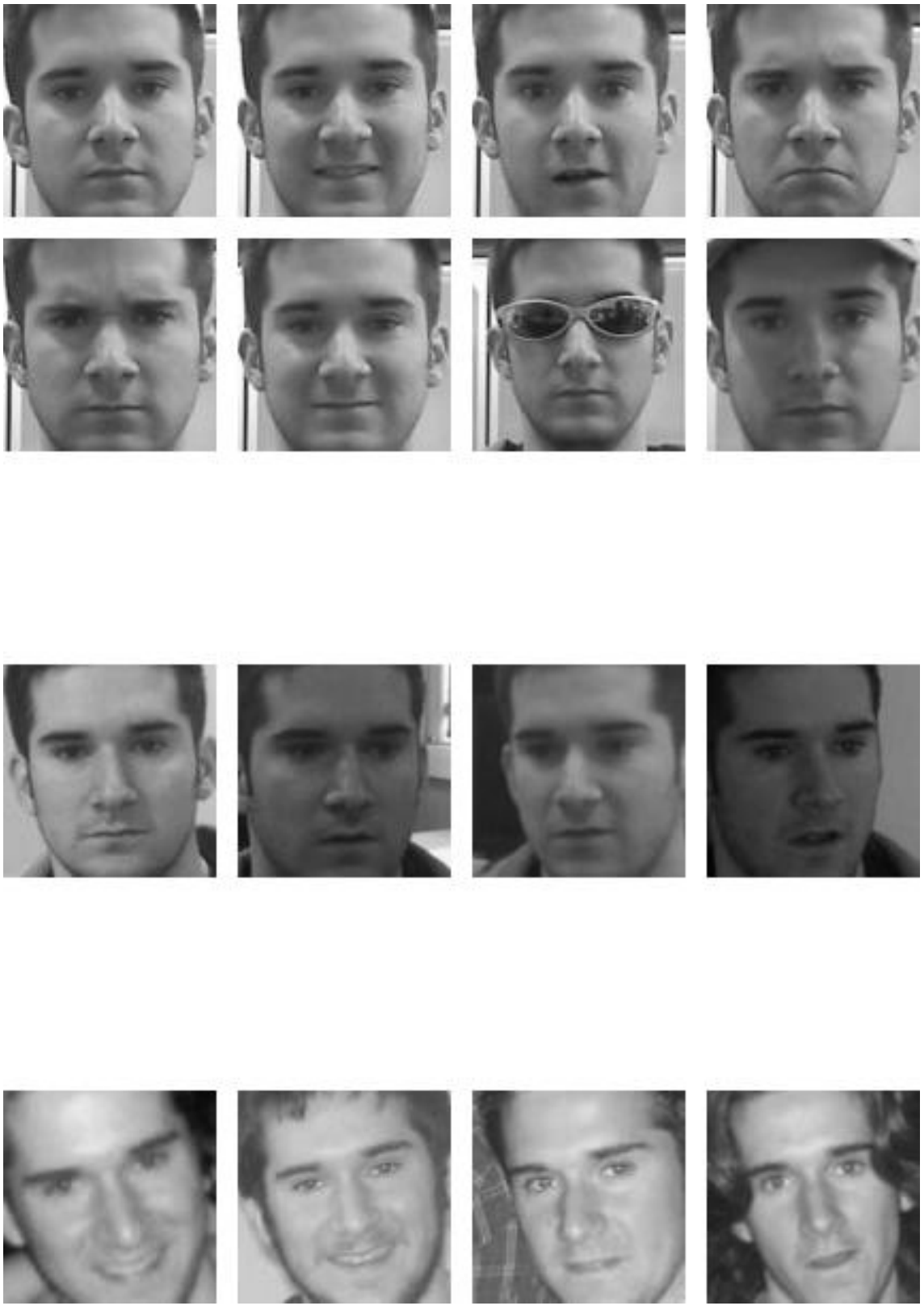

Fig. 6.10 – Conjuntos de imágenes sujeto 10 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

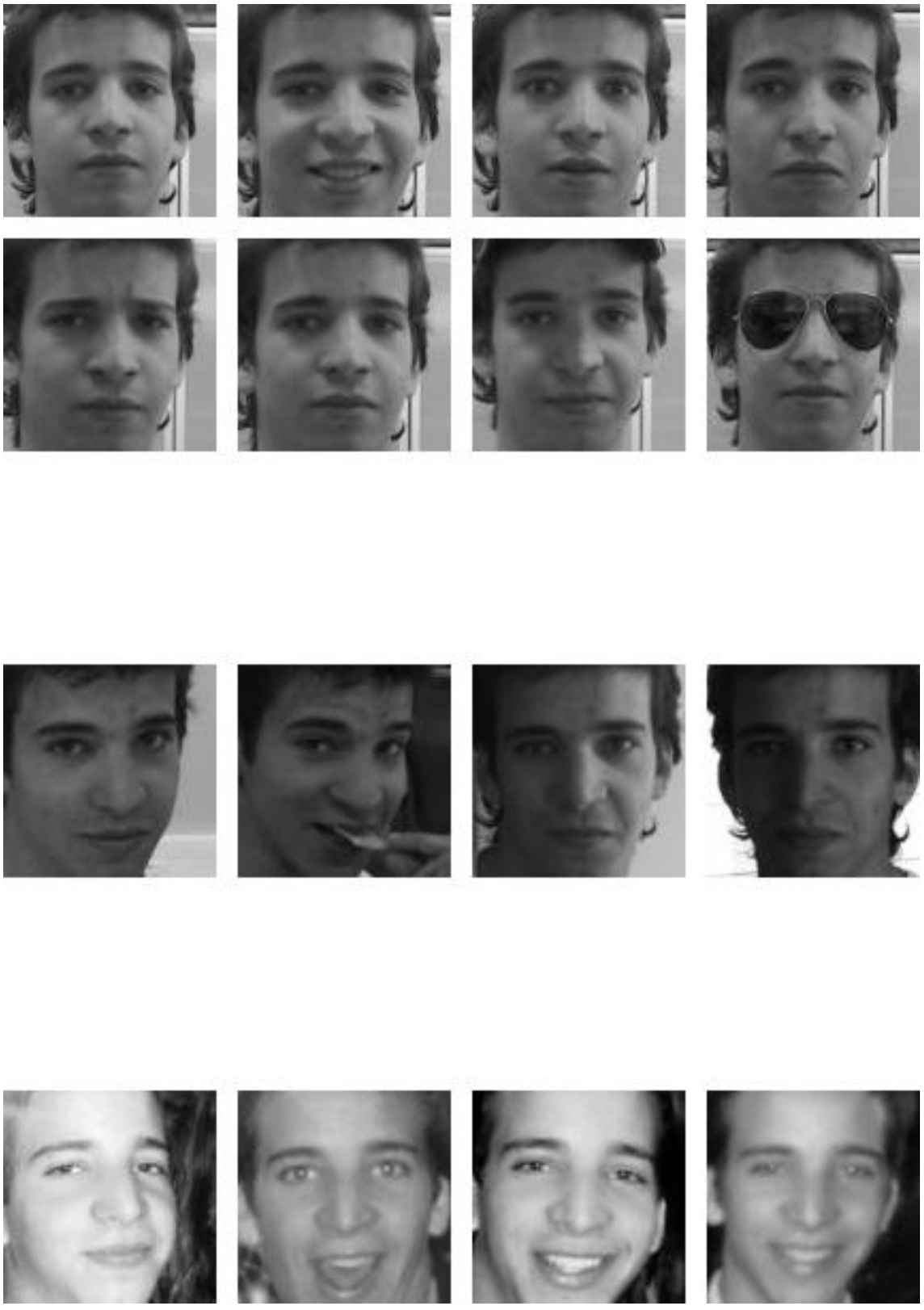

Fig. 6.11 – Conjuntos de imágenes sujeto 11 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

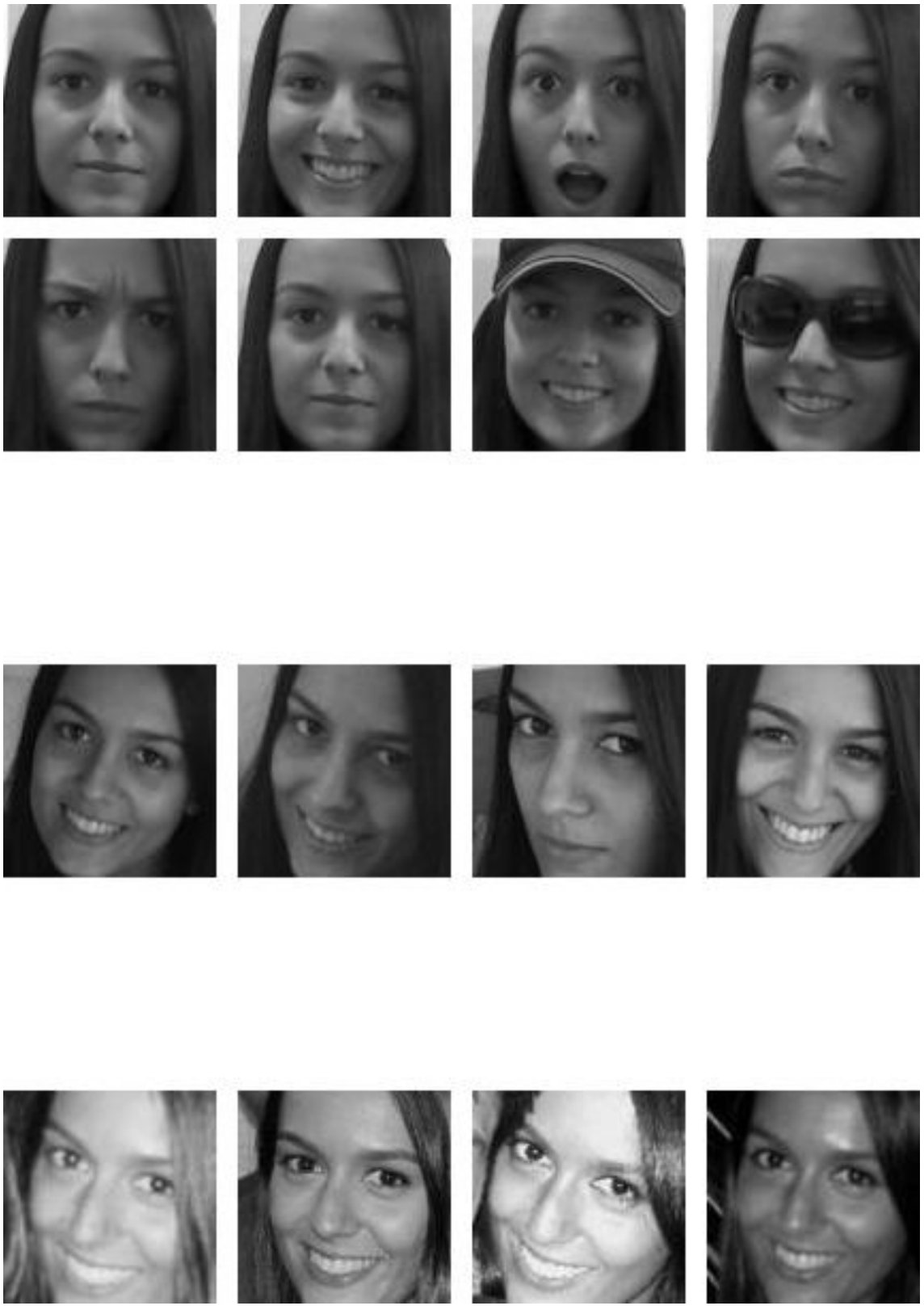

Fig. 6.12 – Conjuntos de imágenes sujeto 12 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.13 – Conjuntos de imágenes sujeto 13 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.14 – Conjuntos de imágenes sujeto 14 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.15 – Conjuntos de imágenes sujeto 15 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

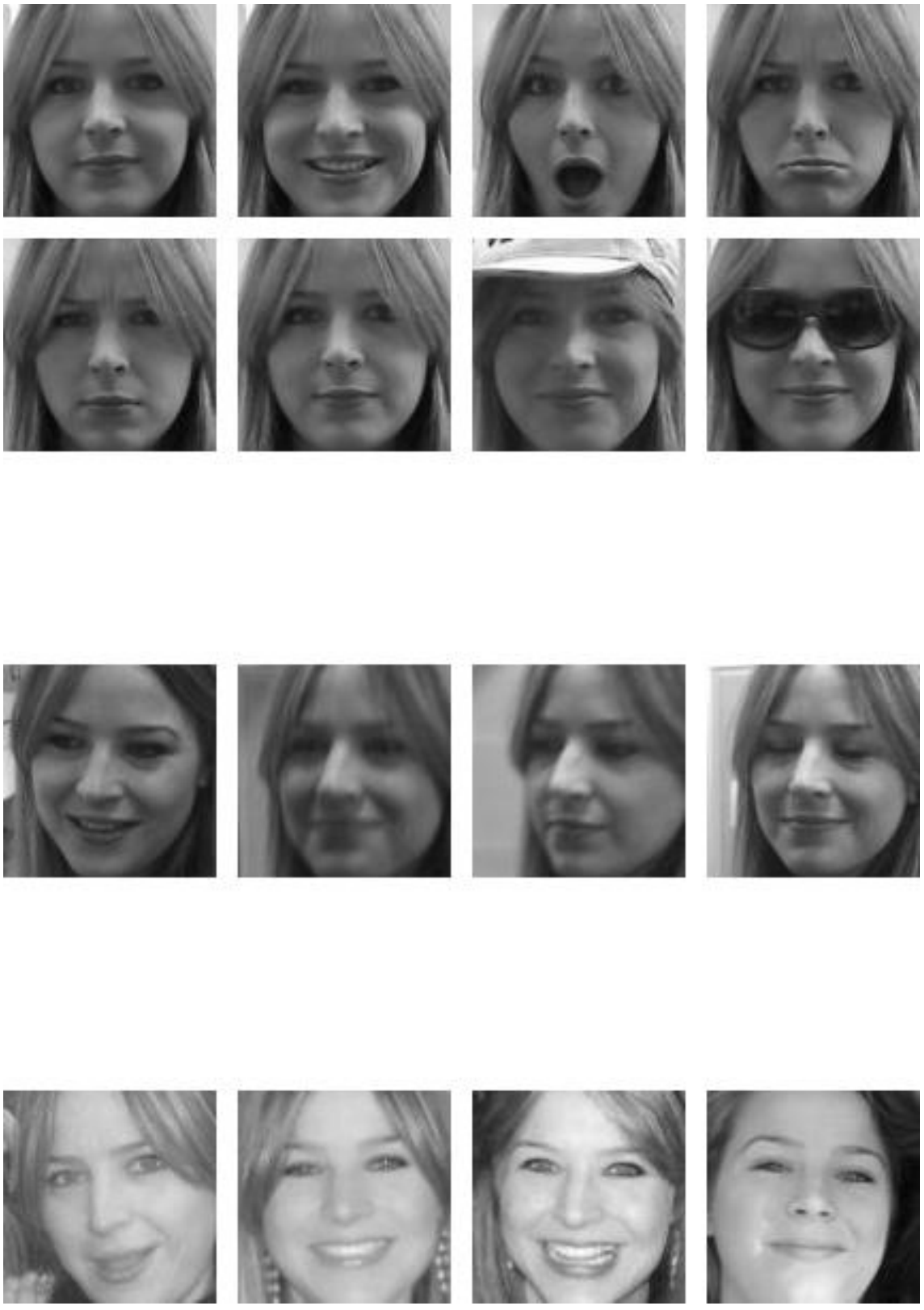

Fig. 6.16 – Conjuntos de imágenes sujeto 16 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.17 – Conjuntos de imágenes sujeto 17 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

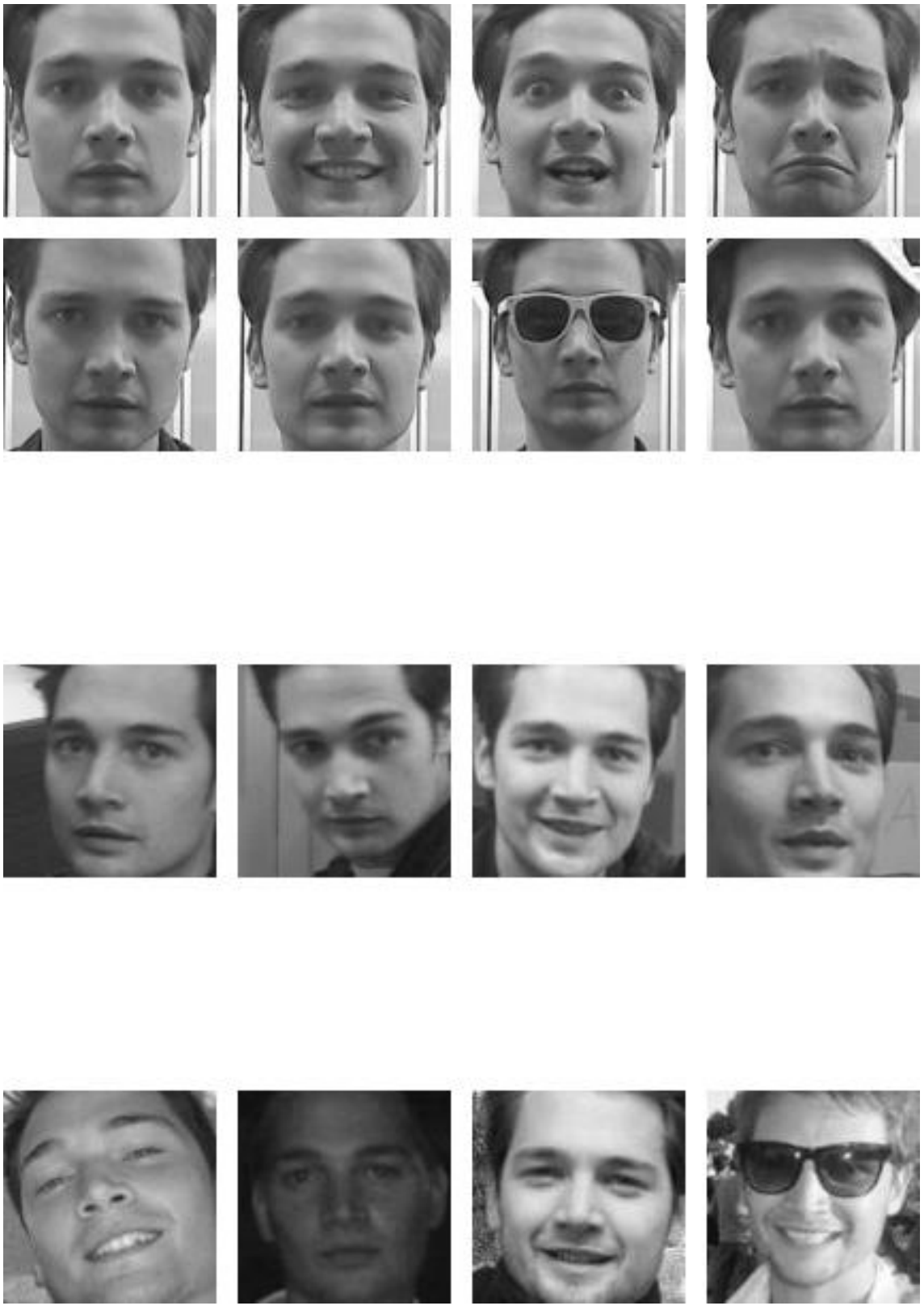

Fig. 6.18 – Conjuntos de imágenes sujeto 18 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

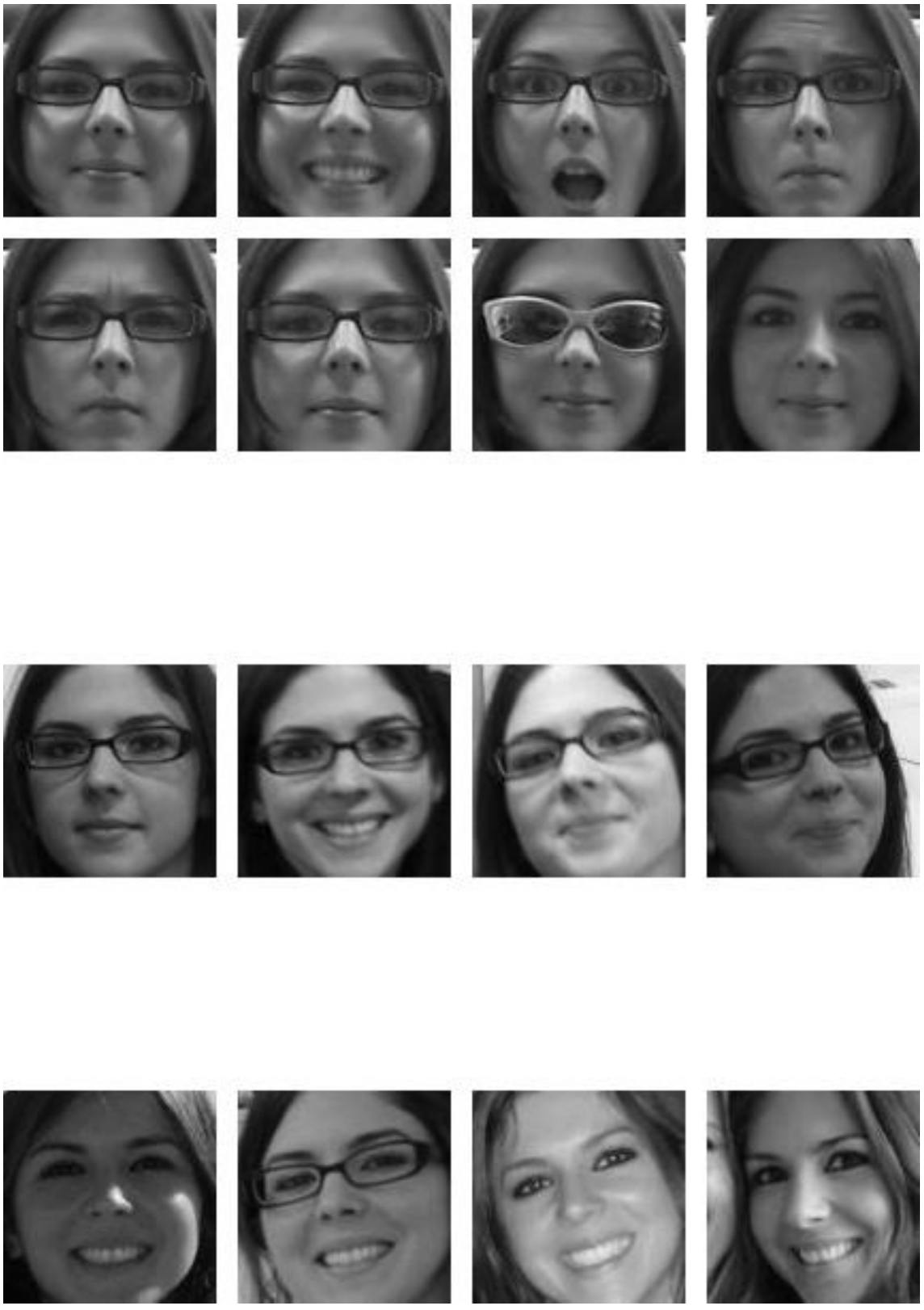

Fig. 6.19 – Conjuntos de imágenes sujeto 19 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.20 – Conjuntos de imágenes sujeto 20 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

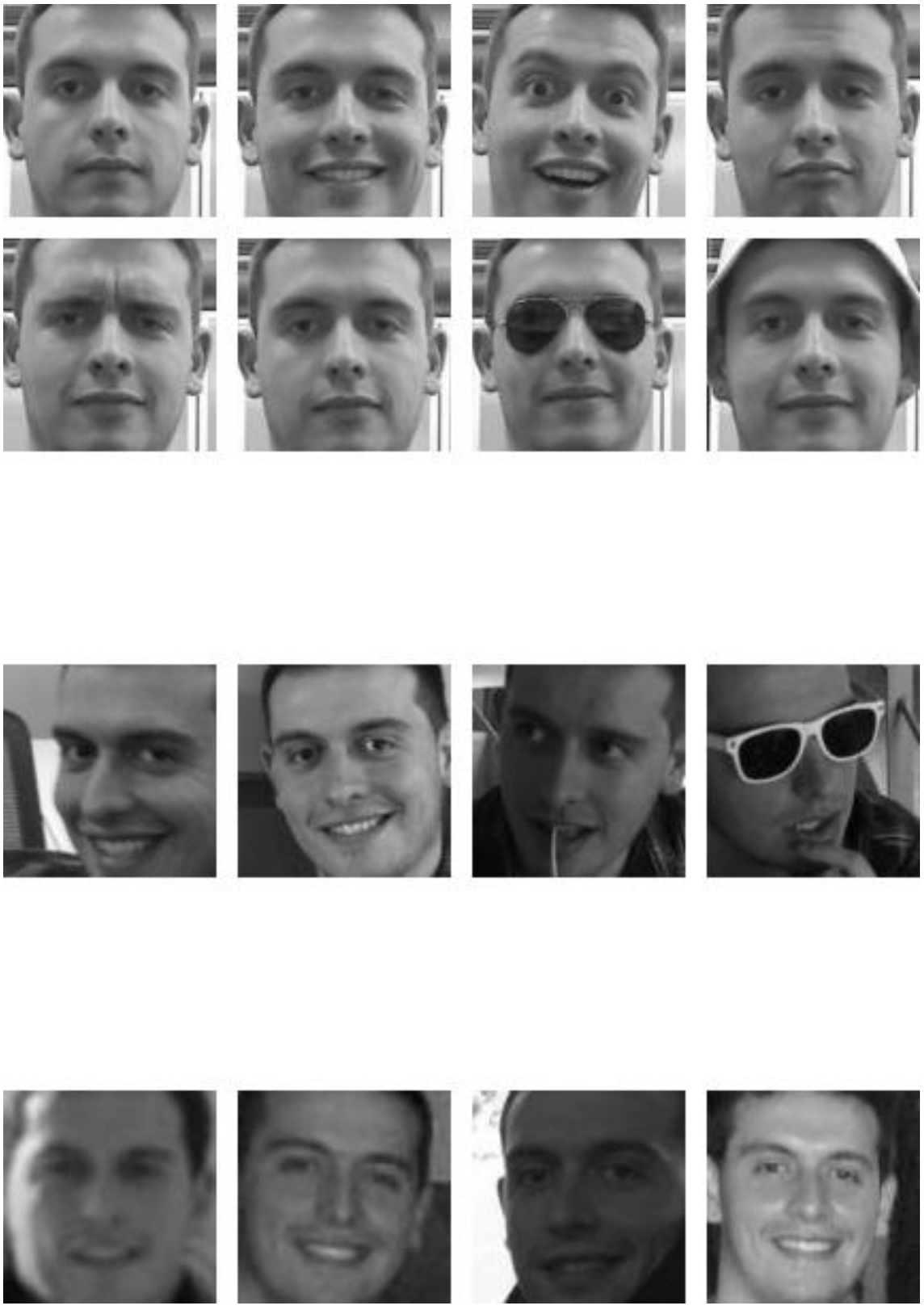

Fig. 6.21 – Conjuntos de imágenes sujeto 21 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.22 – Conjuntos de imágenes sujeto 22 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.23 – Conjuntos de imágenes sujeto 23 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.24 – Conjuntos de imágenes sujeto 24 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.25 – Conjuntos de imágenes sujeto 25 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.26 – Conjuntos de imágenes sujeto 26 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.27 – Conjuntos de imágenes sujeto 27 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

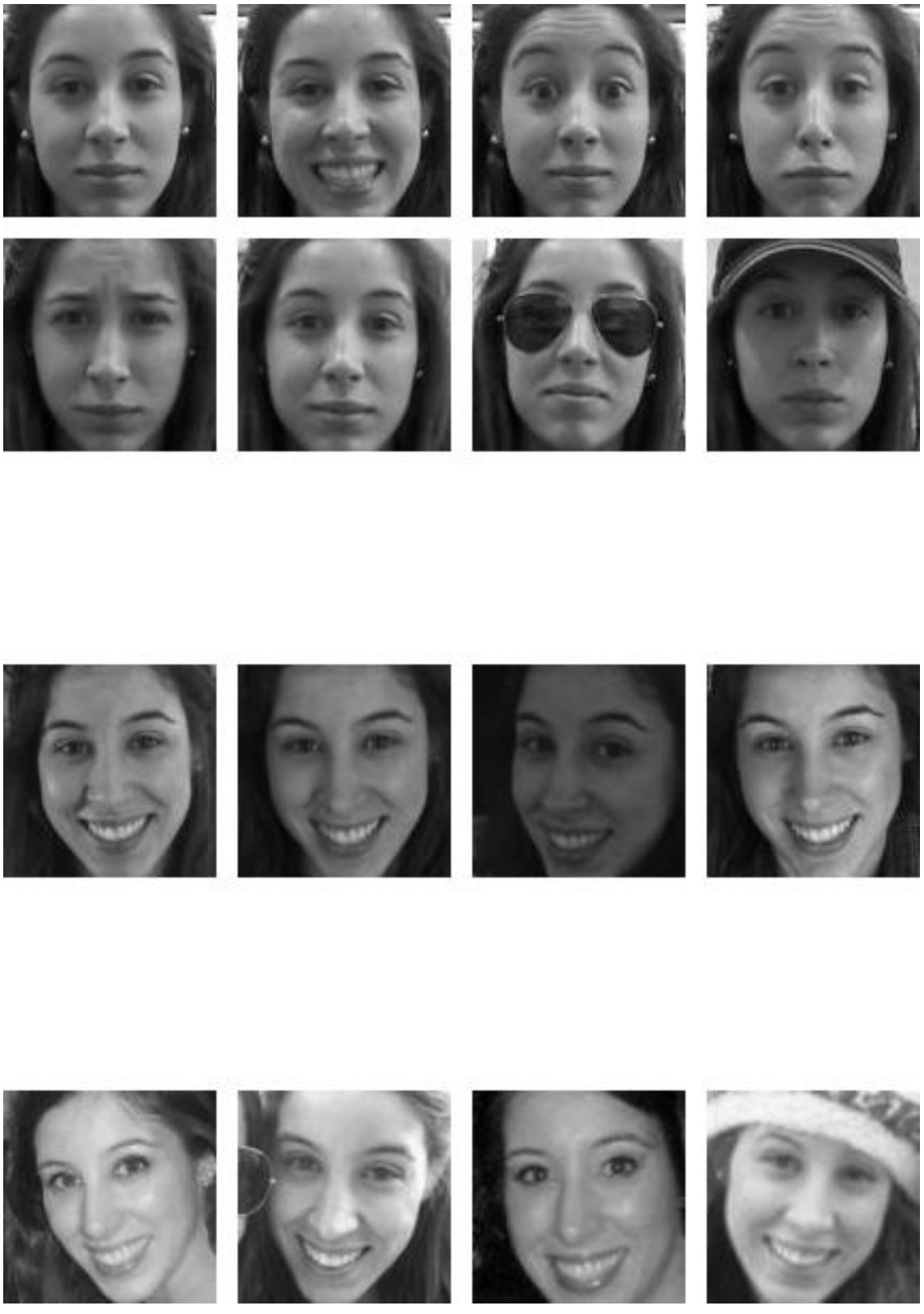

Fig. 6.28 – Conjuntos de imágenes sujeto 28 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

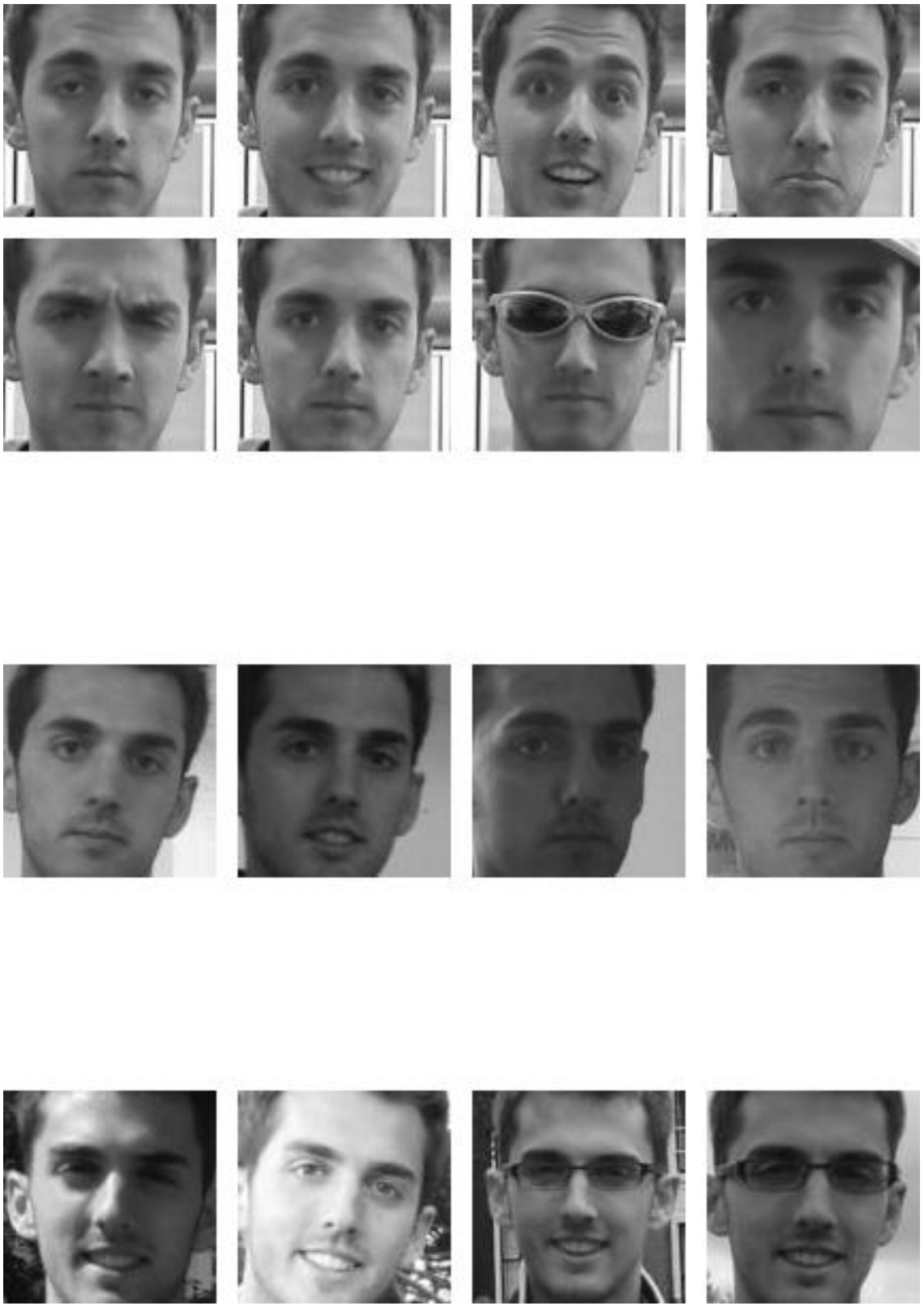

Fig. 6.29 – Conjuntos de imágenes sujeto 29 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

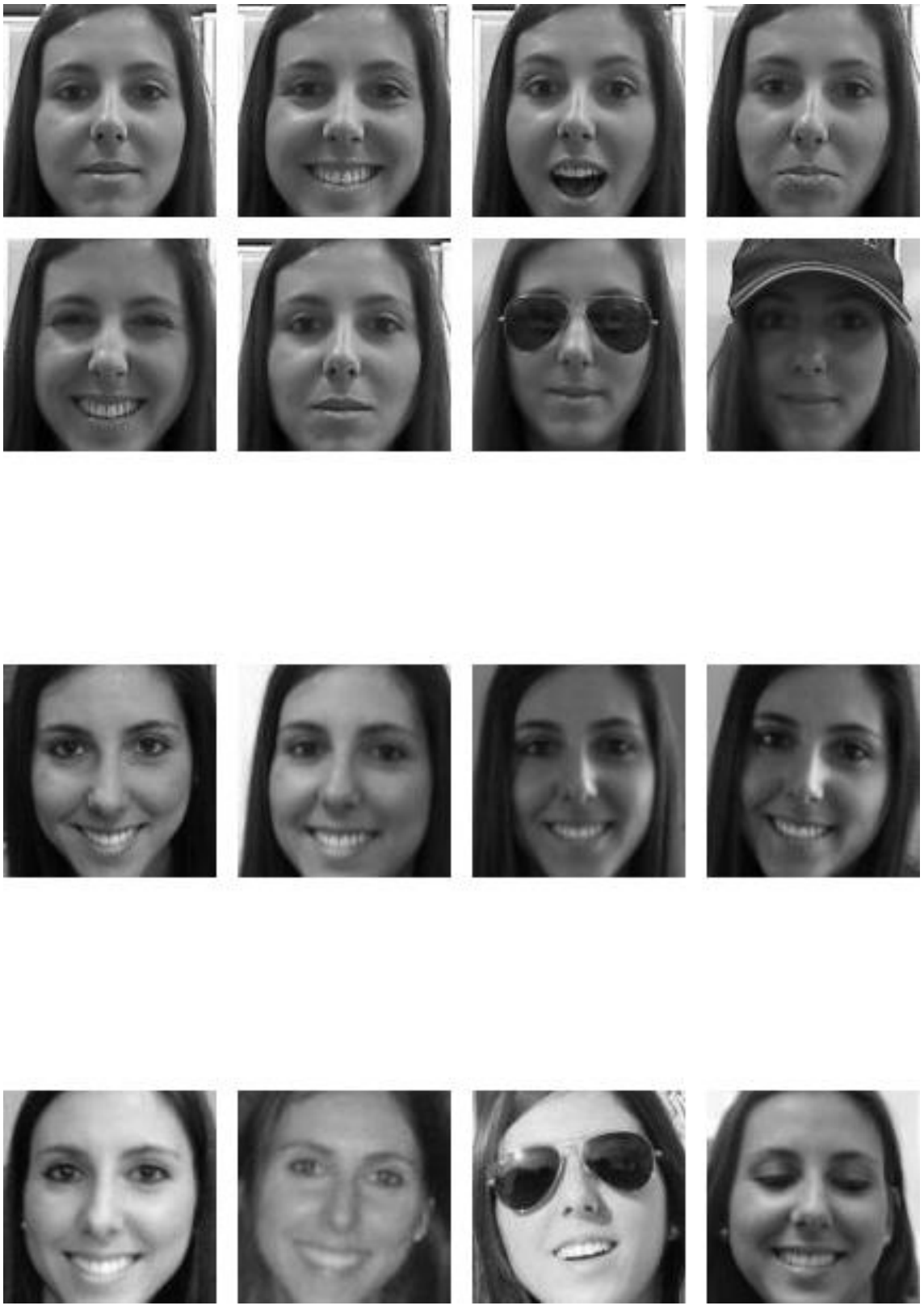

Fig.6.30 – Conjuntos de imágenes sujeto 30 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig.6.31 – Conjuntos de imágenes sujeto 31 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.32 – Conjuntos de imágenes sujeto 32 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.33– Conjuntos de imágenes sujeto 33 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.34 – Conjuntos de imágenes sujeto 34 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.35 – Conjuntos de imágenes sujeto 35 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

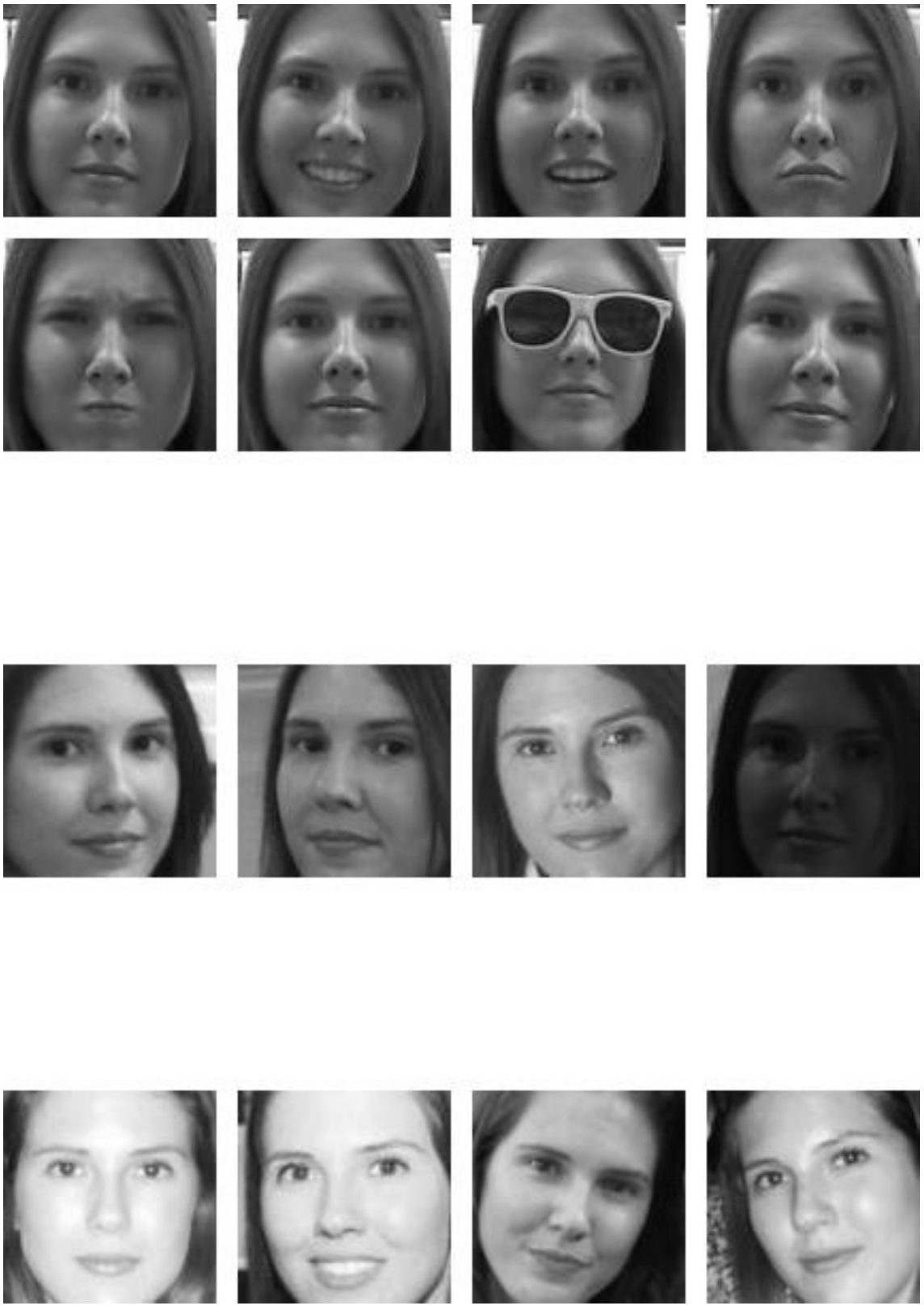

Fig. 6.36 – Conjuntos de imágenes sujeto 36 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

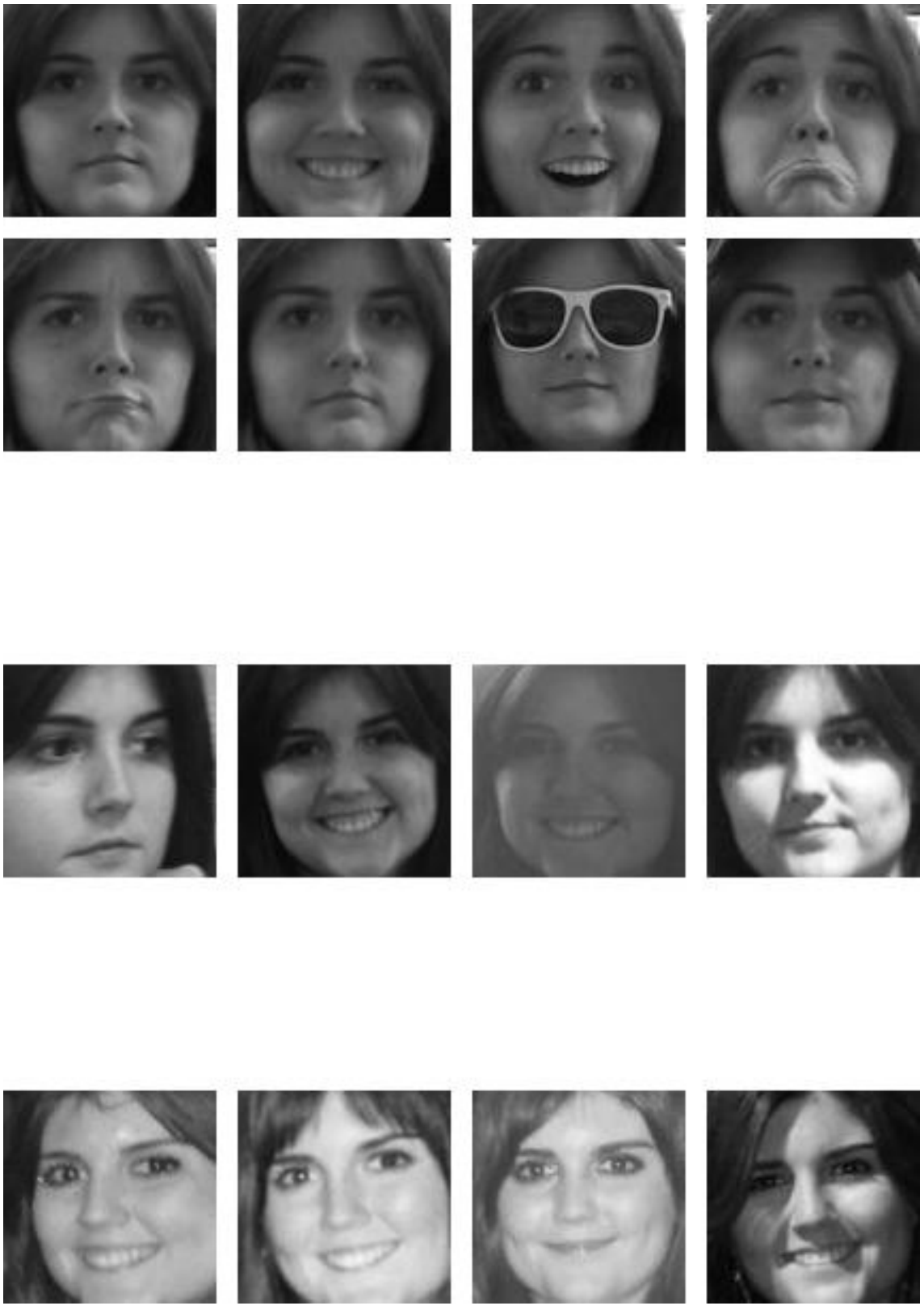

Fig. 6.37 – Conjuntos de imágenes sujeto 37 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.38 – Conjuntos de imágenes sujeto 38 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

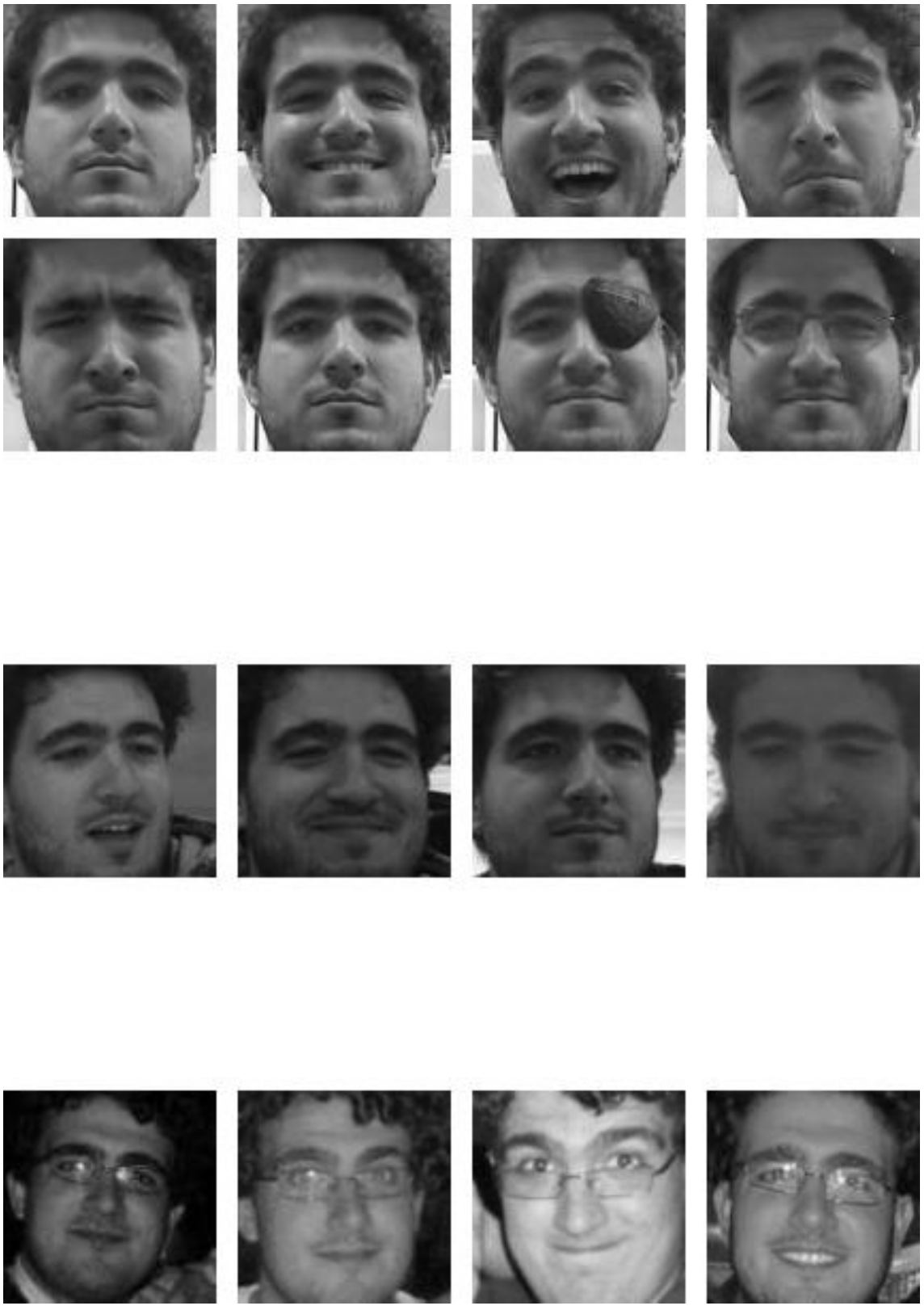

Fig. 6.39 – Conjuntos de imágenes sujeto 39 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

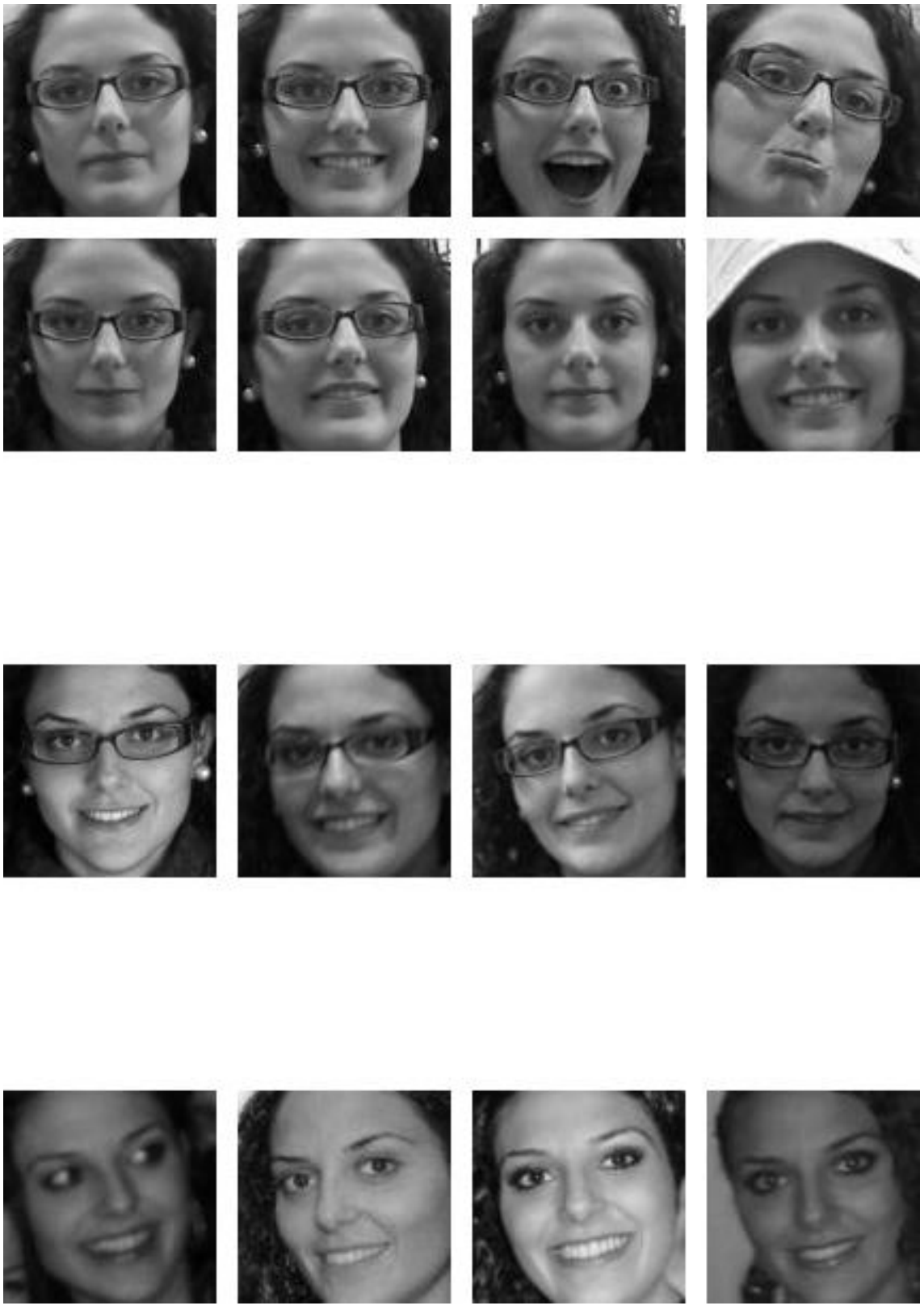

Fig. 6.40 – Conjuntos de imágenes sujeto 40 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

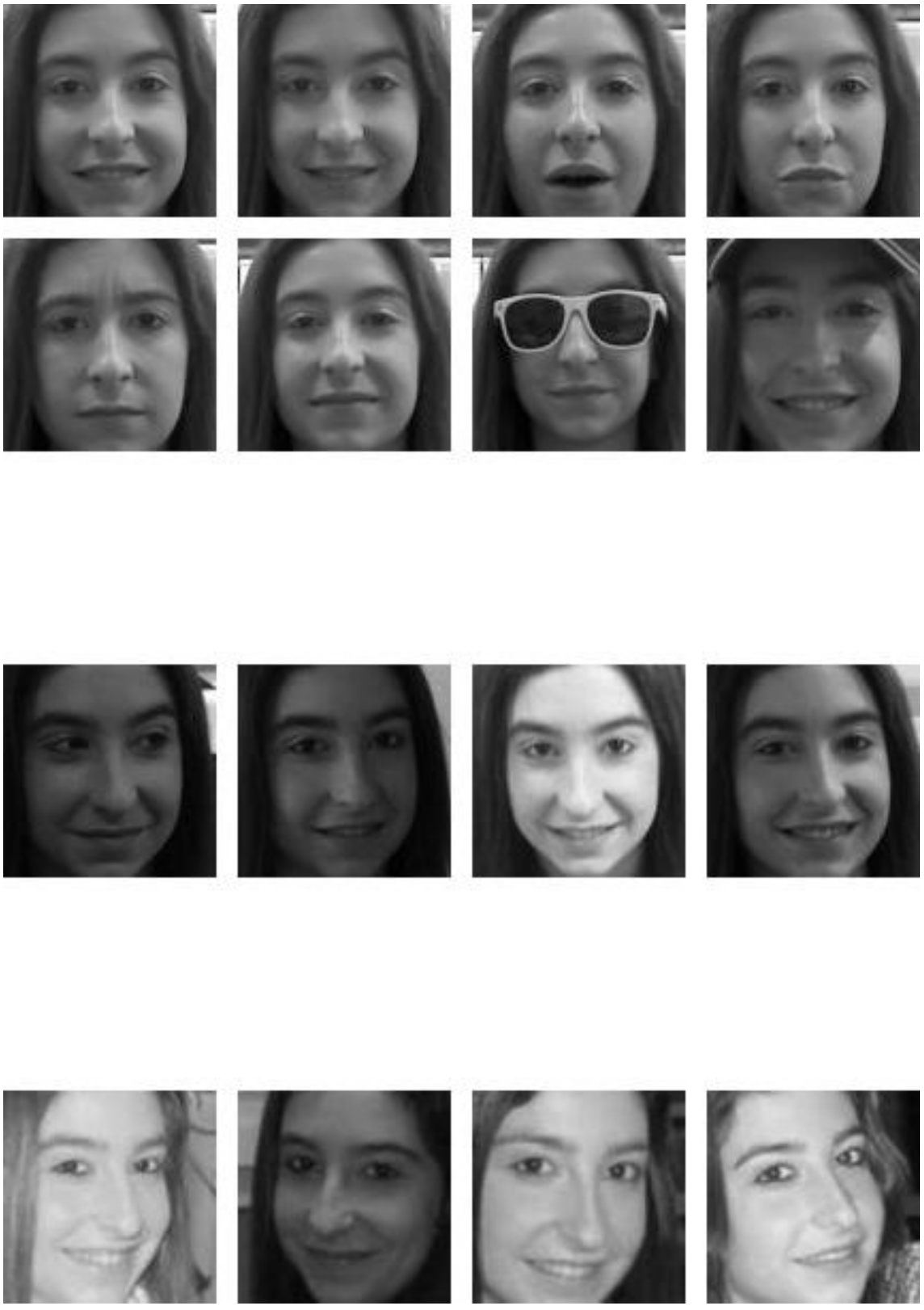

Fig. 6.41 – Conjuntos de imágenes sujeto 41 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.42 – Conjuntos de imágenes sujeto 42 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.43 – Conjuntos de imágenes sujeto 43 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

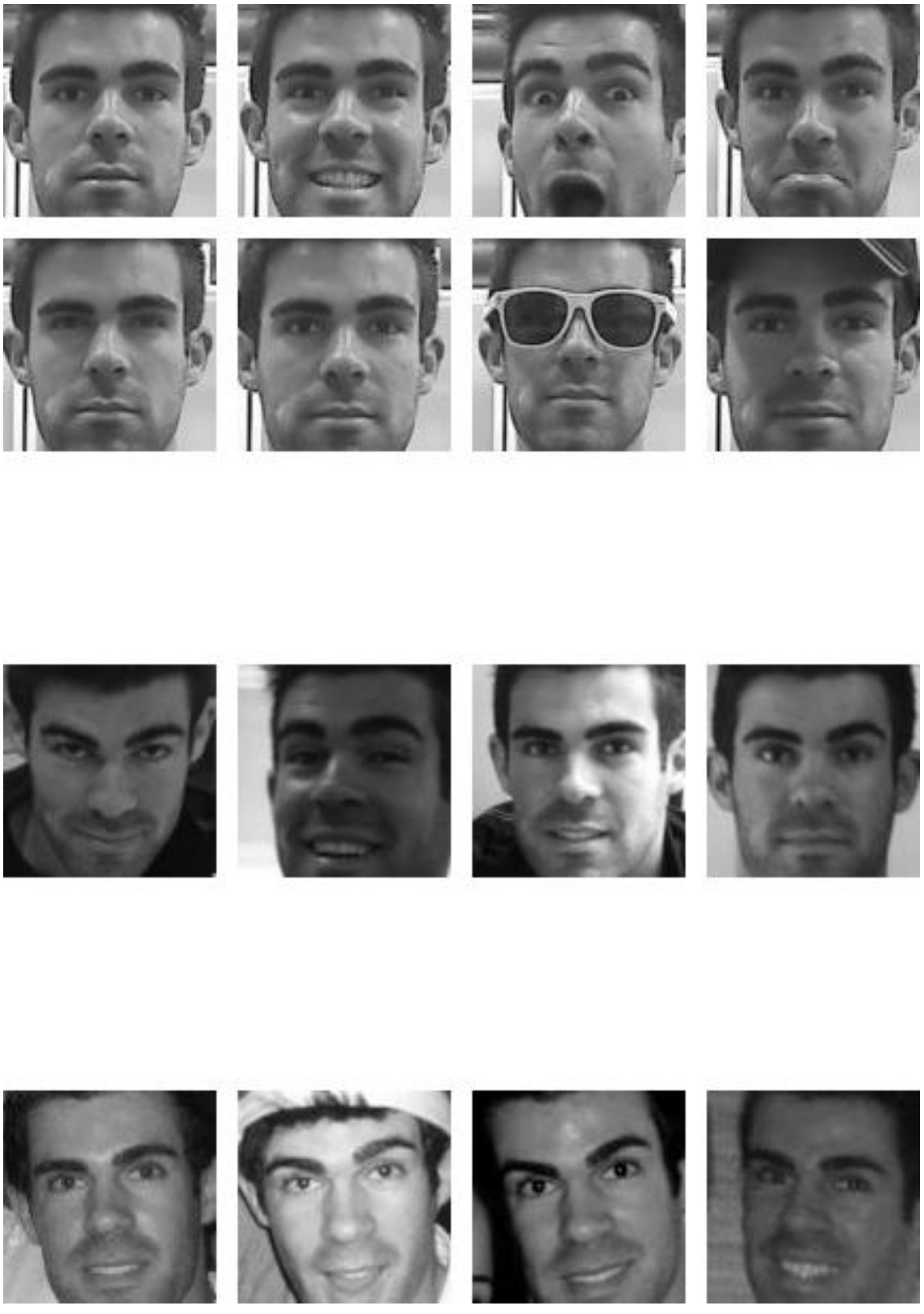

Fig. 6.44 – Conjuntos de imágenes sujeto 44 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.45 – Conjuntos de imágenes sujeto 45 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

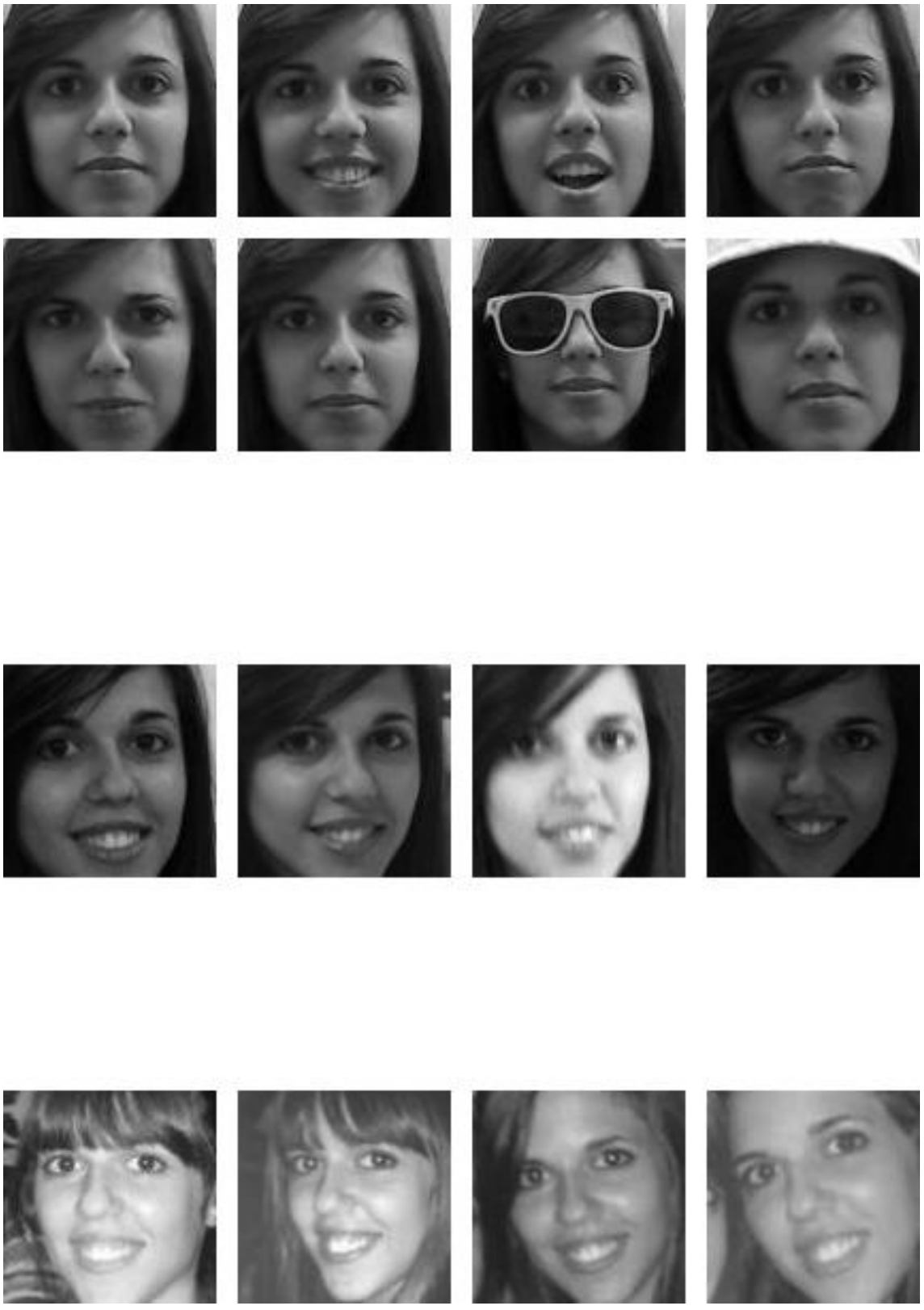

Fig. 6.46 – Conjuntos de imágenes sujeto 46 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

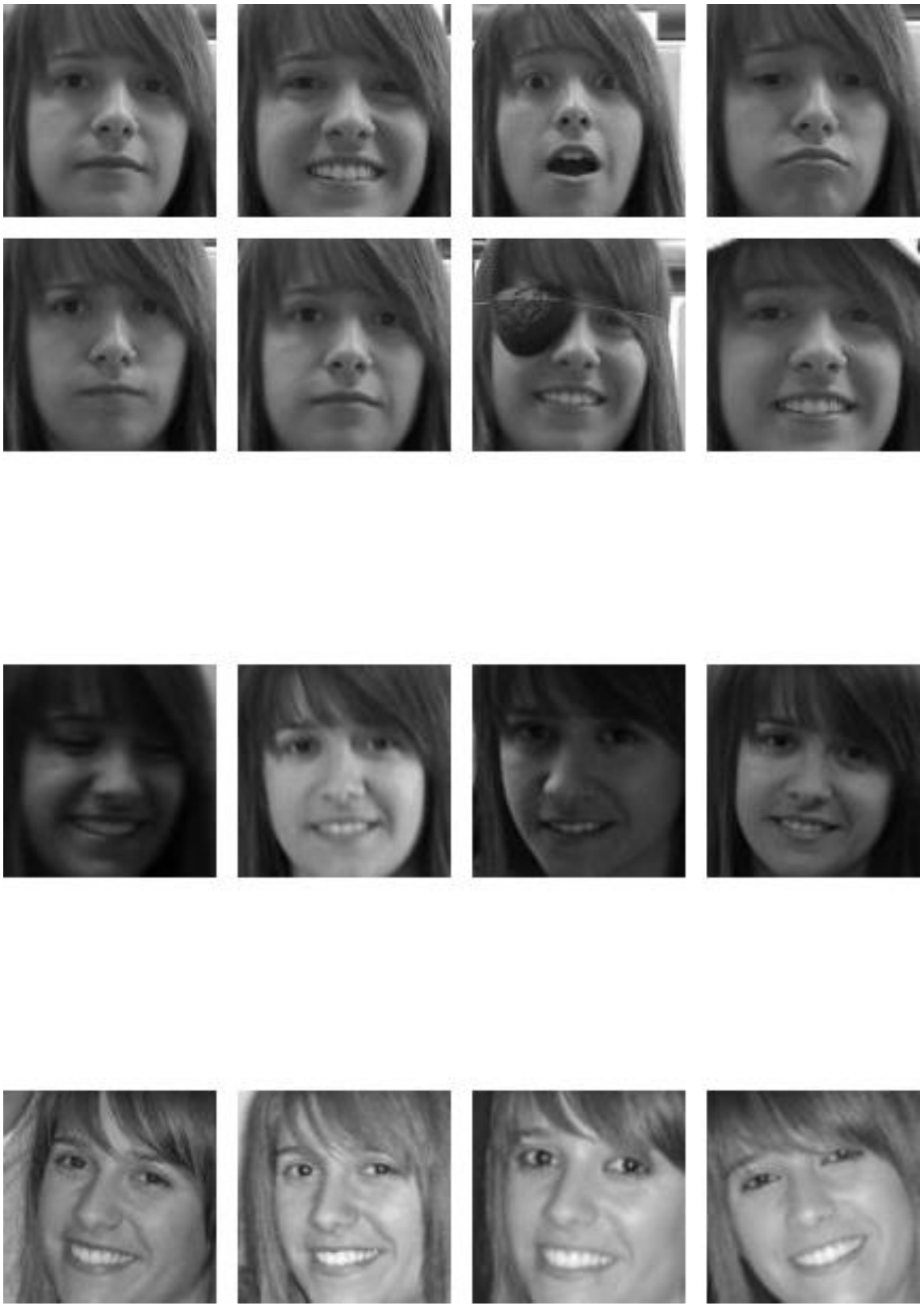

Fig. 6.47 – Conjuntos de imágenes sujeto 47 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

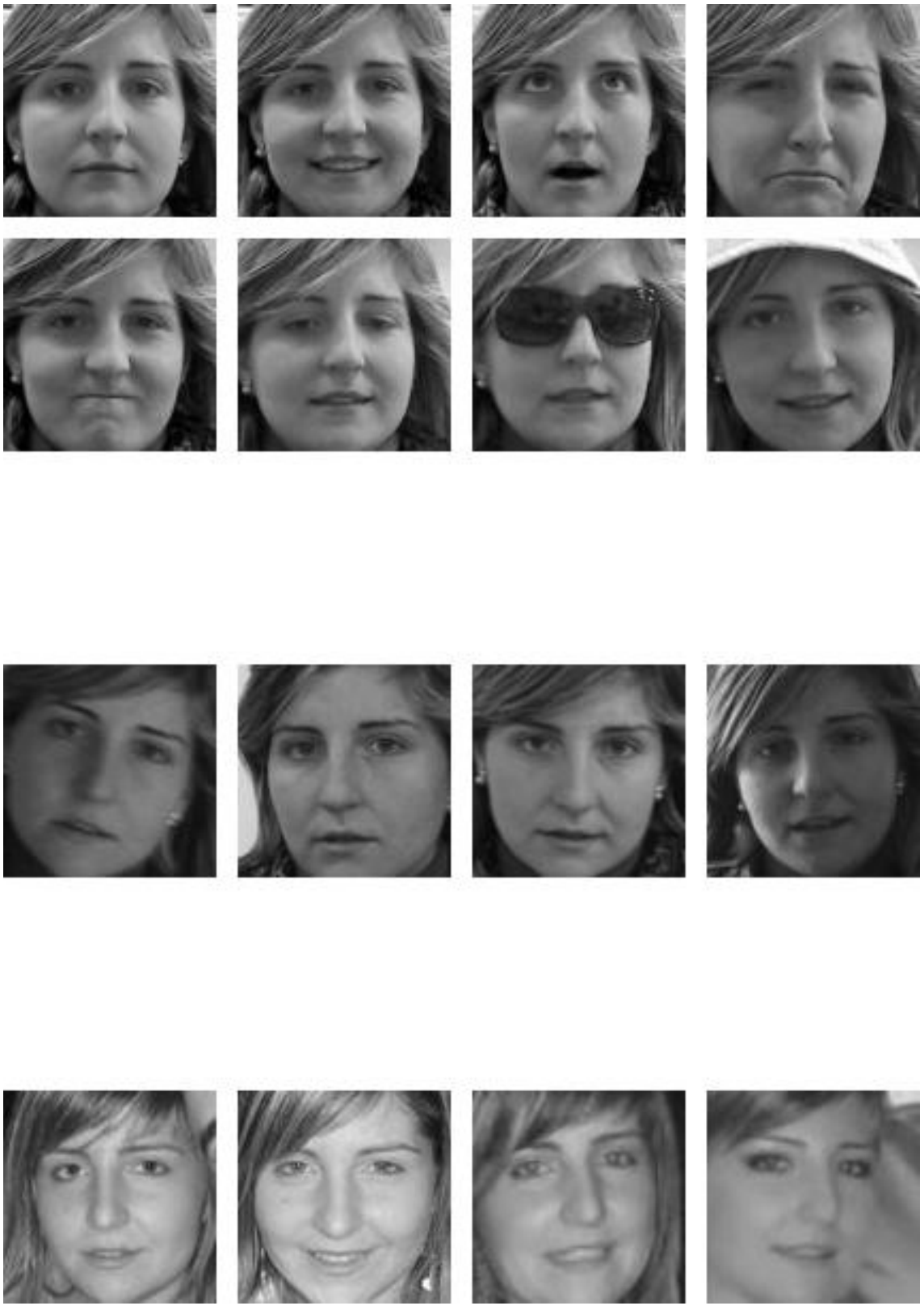

Fig. 6.48 – Conjuntos de imágenes sujeto 48 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

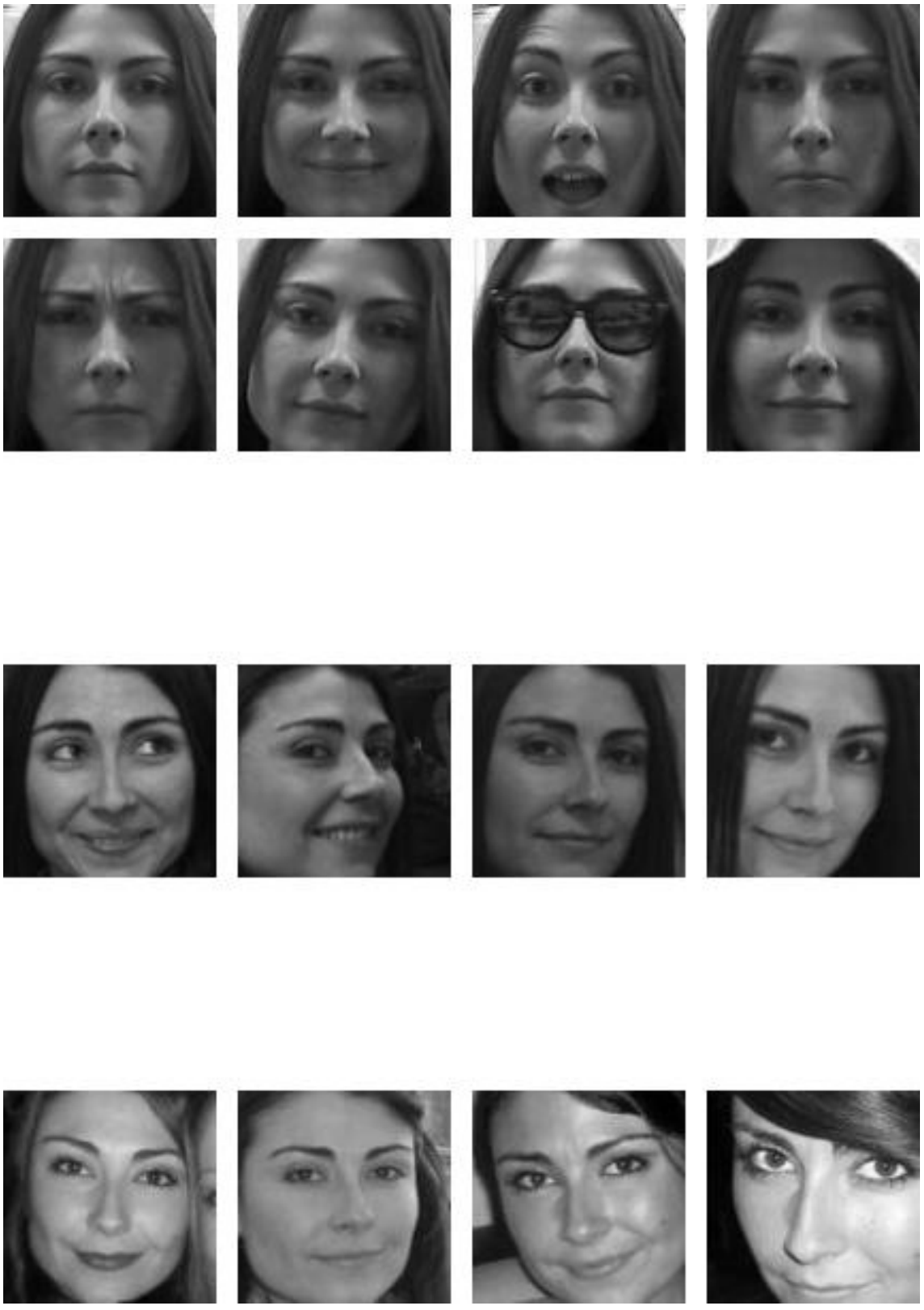

Fig. 6.49 – Conjuntos de imágenes sujeto 49 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

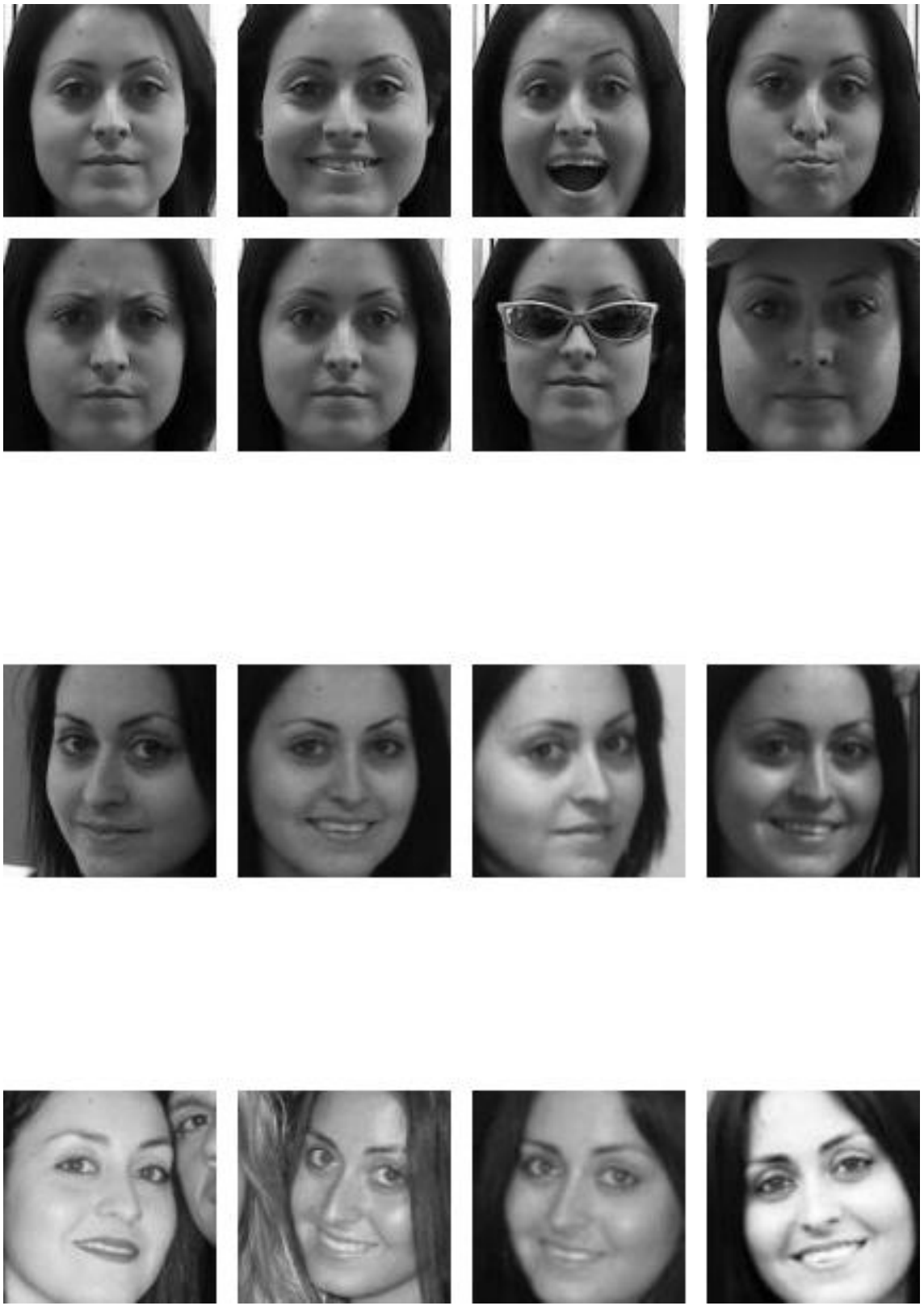

Fig. 6.50 – Conjuntos de imágenes sujeto 50 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.51 – Conjuntos de imágenes sujeto 51 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.52 – Conjuntos de imágenes sujeto 52 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

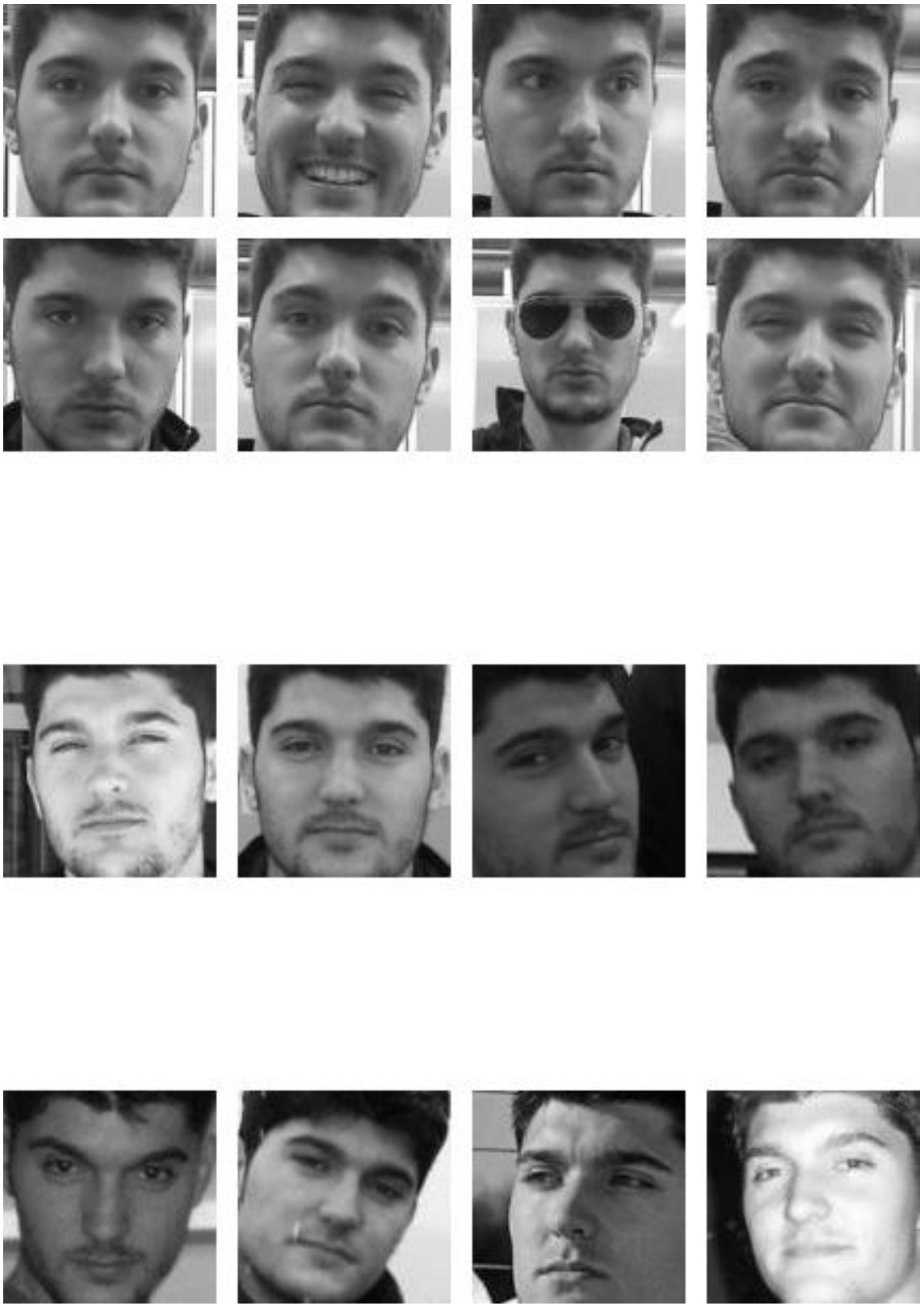

Fig. 6.53 – Conjuntos de imágenes sujeto 53 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.54 – Conjuntos de imágenes sujeto 54 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

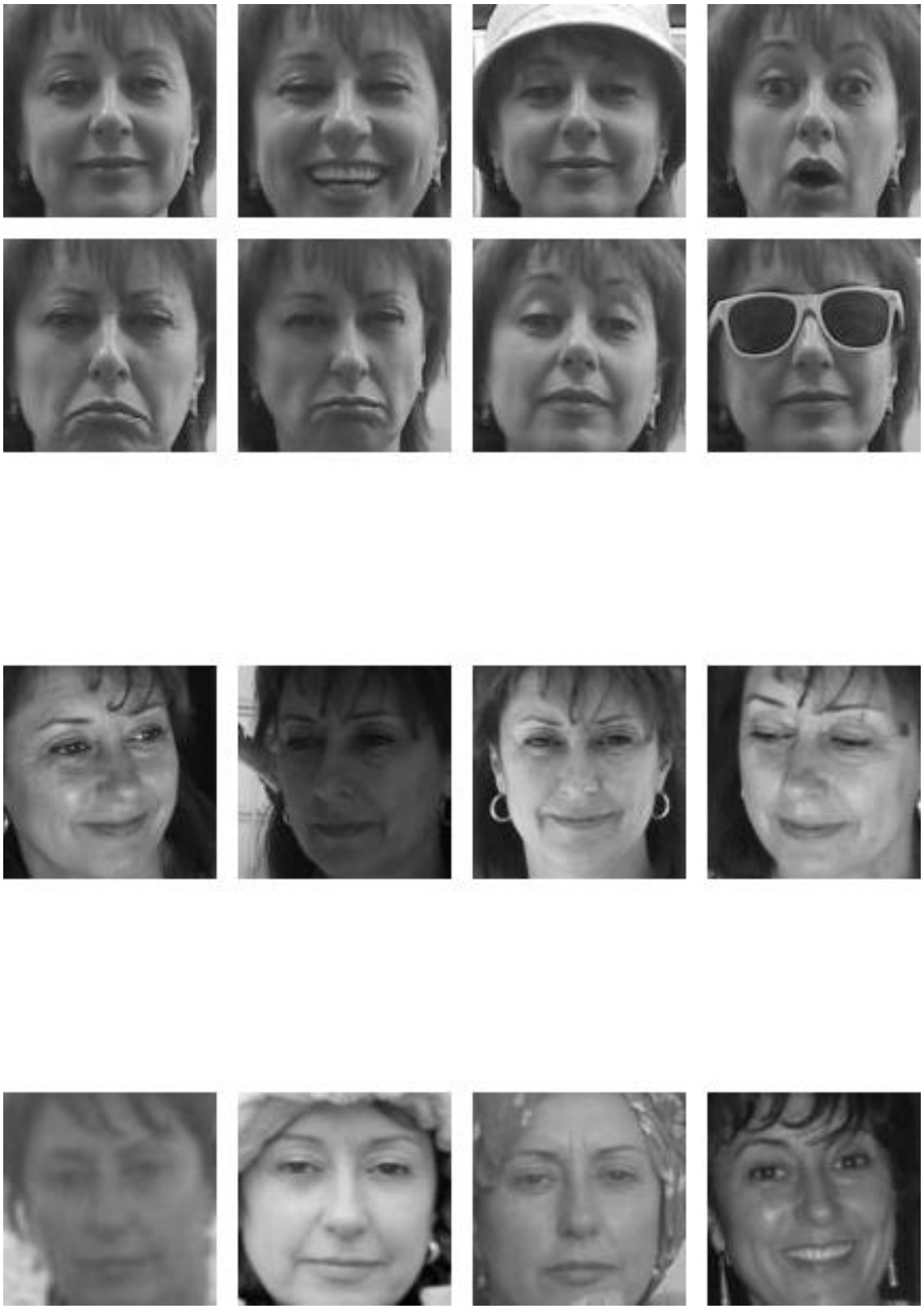

Fig. 6.55 – Conjuntos de imágenes sujeto 55 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.56 – Conjuntos de imágenes sujeto 56 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

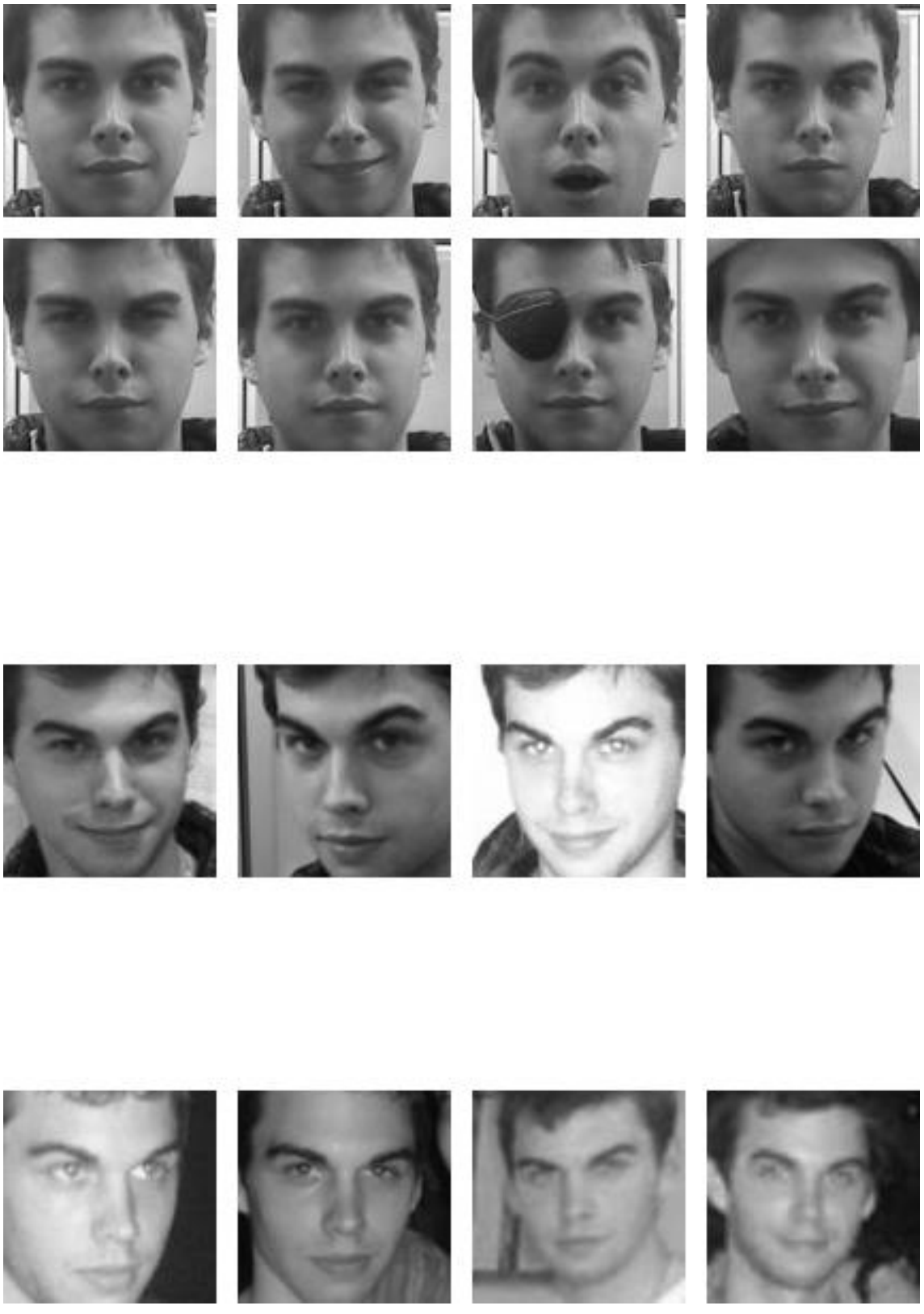

Fig. 6.57 – Conjuntos de imágenes sujeto 57 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

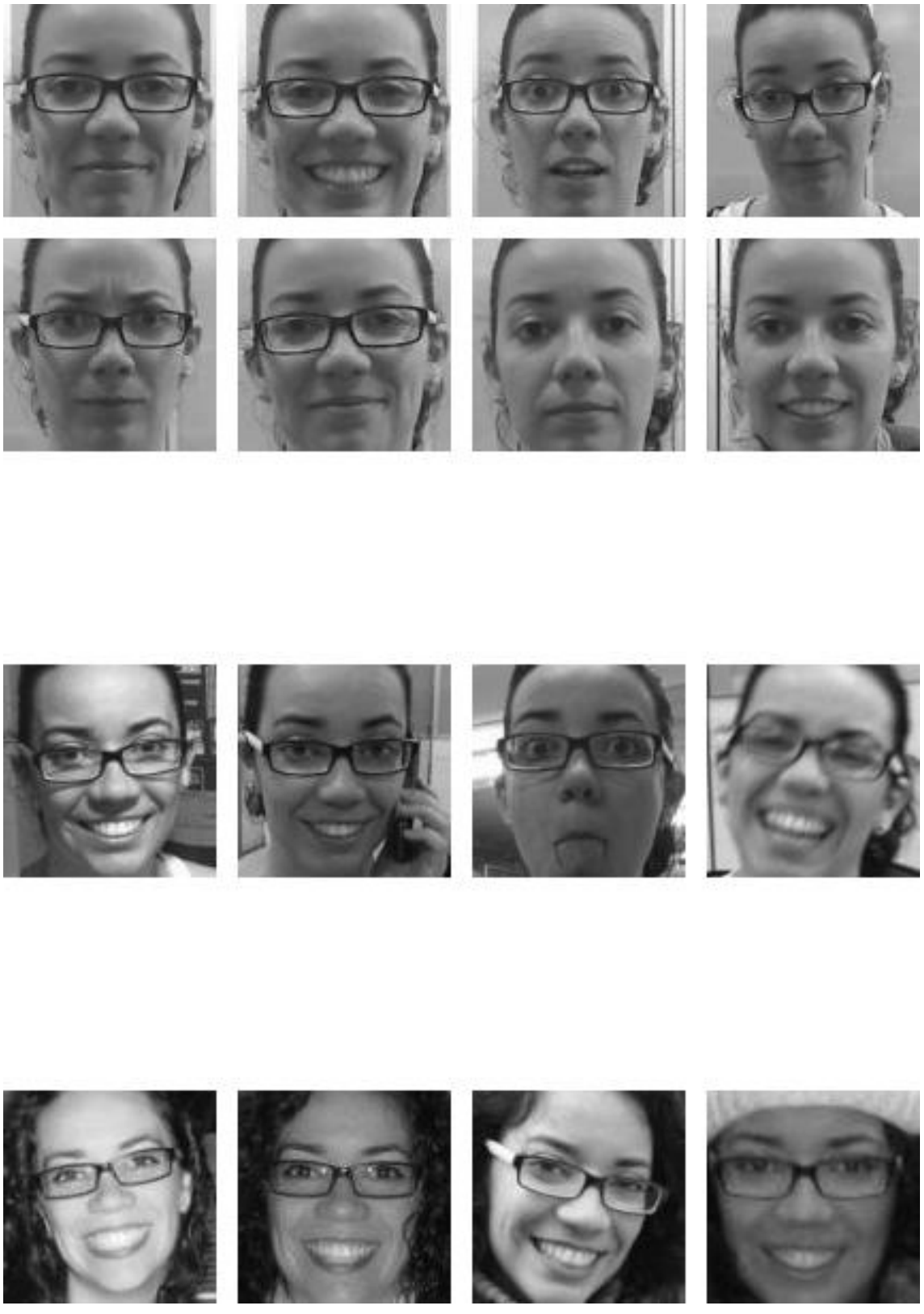

Fig. 6.58 – Conjuntos de imágenes sujeto 58 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

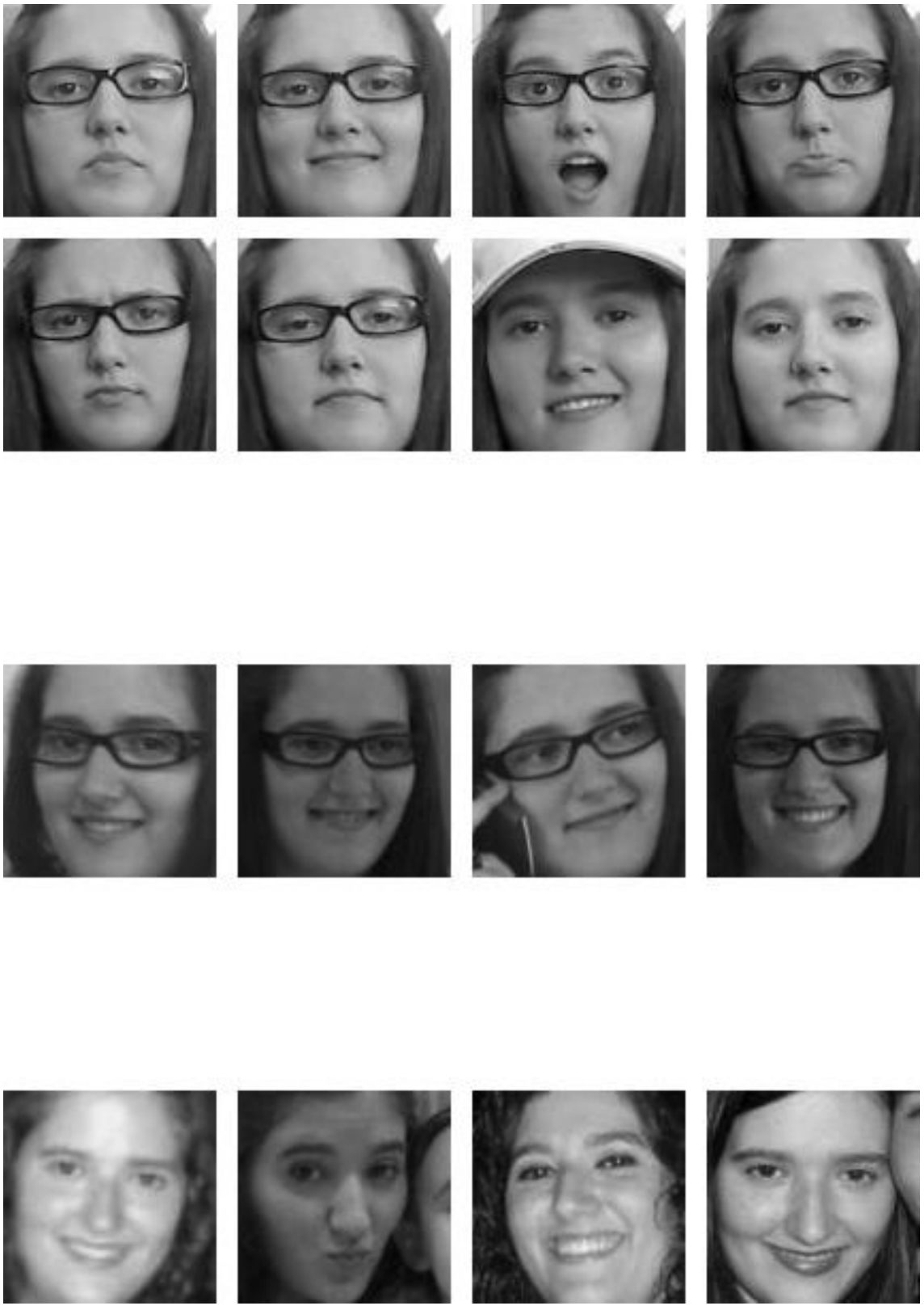

Fig. 6.59 – Conjuntos de imágenes sujeto 59 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.60 – Conjuntos de imágenes sujeto 60 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.61 – Conjuntos de imágenes sujeto 61 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.62 – Conjuntos de imágenes sujeto 62 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

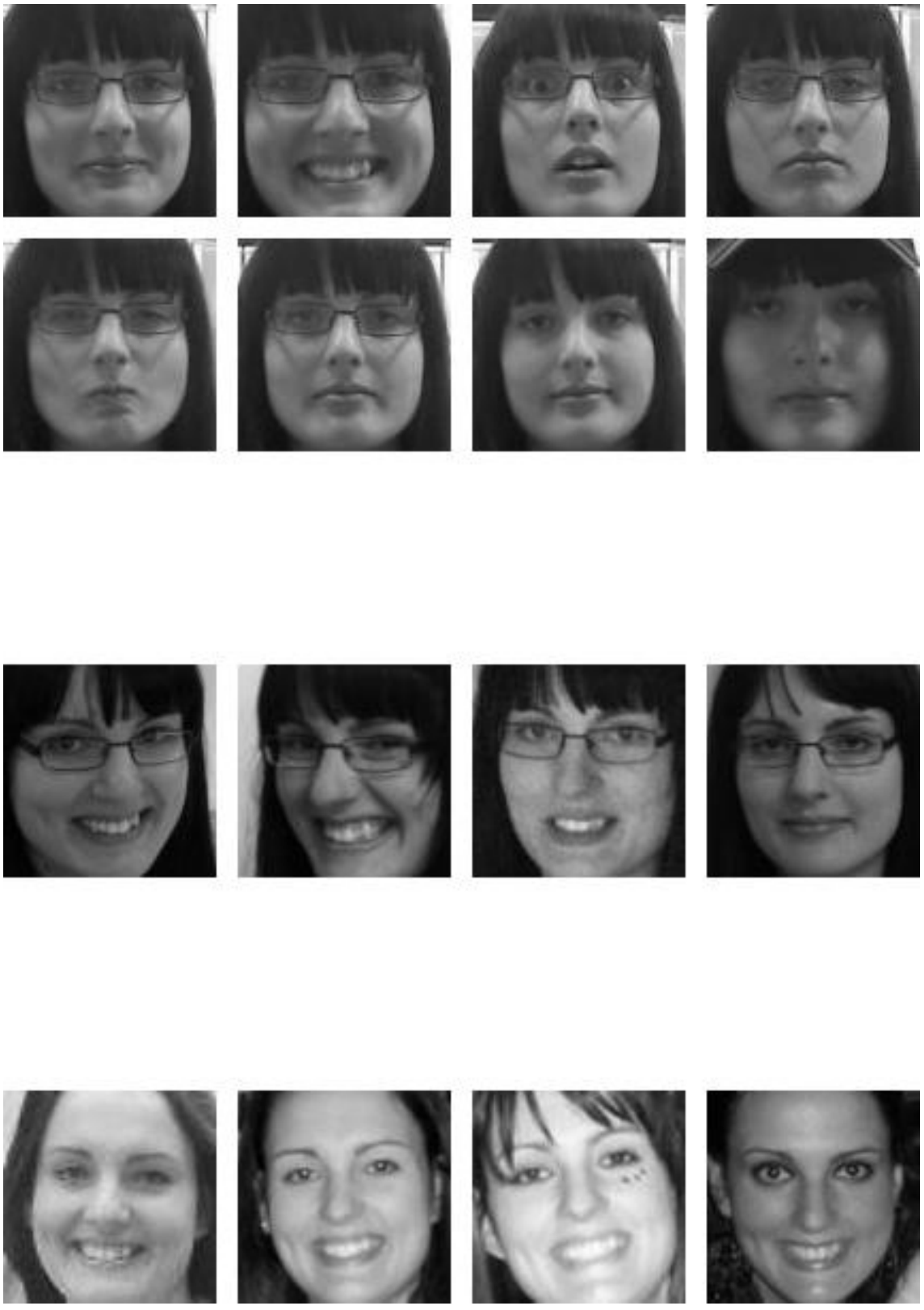

Fig. 6.63 – Conjuntos de imágenes sujeto 63 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.64 – Conjuntos de imágenes sujeto 64 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.65 – Conjuntos de imágenes sujeto 65-

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.66 – Conjuntos de imágenes sujeto 66 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

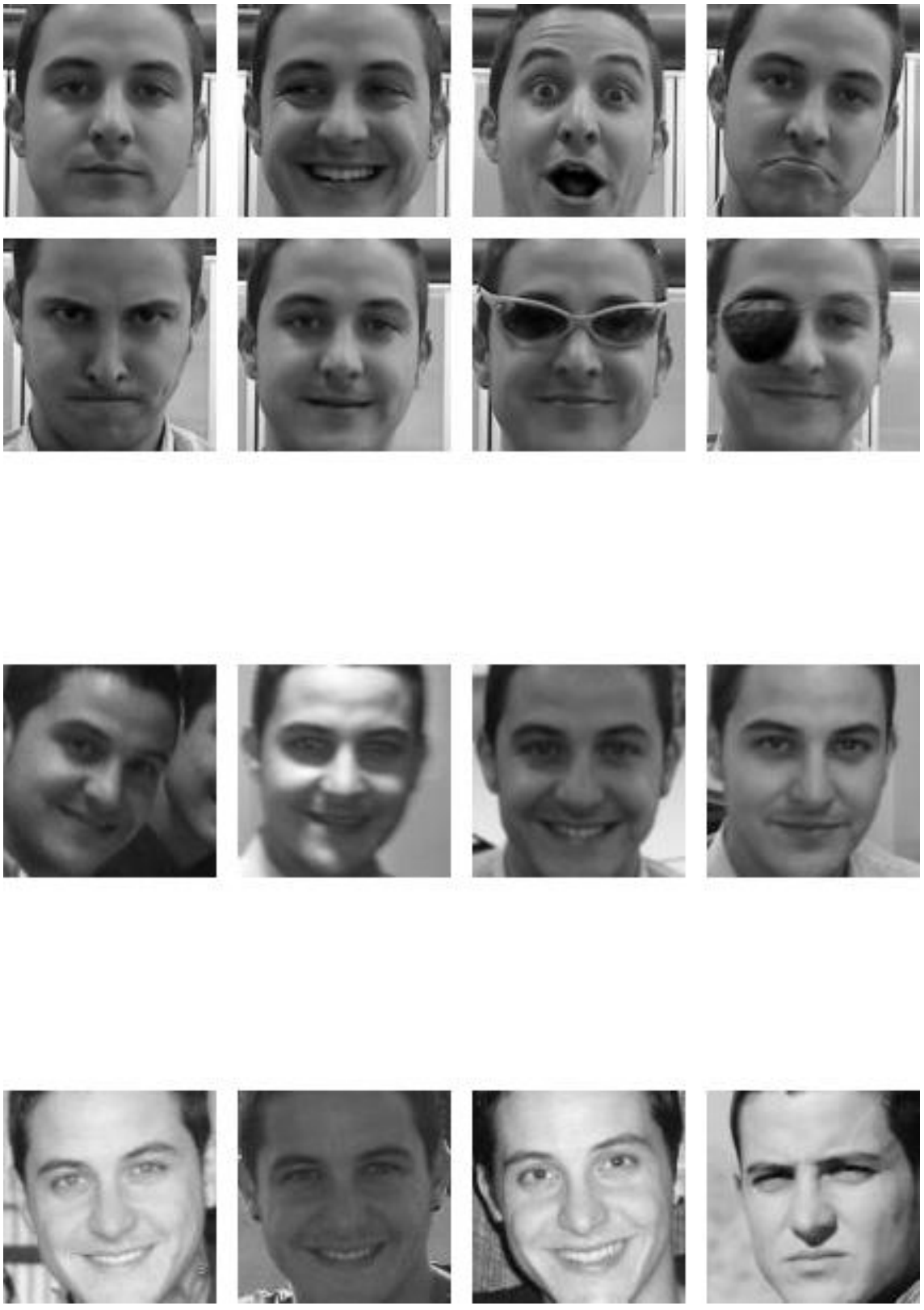

Fig. 6.67 – Conjuntos de imágenes sujeto 67 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.68 – Conjuntos de imágenes sujeto 68 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

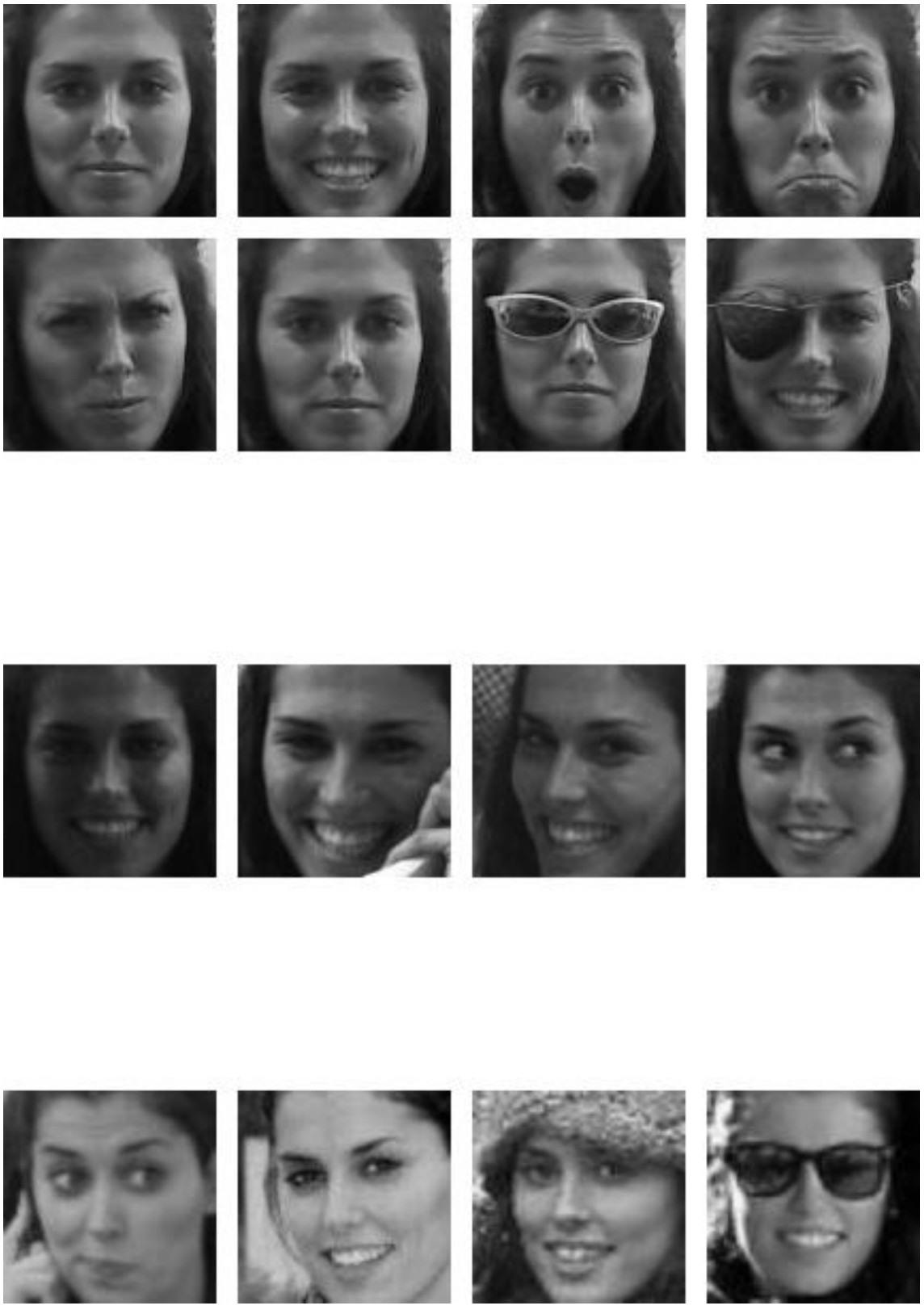

Fig. 6.69 – Conjuntos de imágenes sujeto 69 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

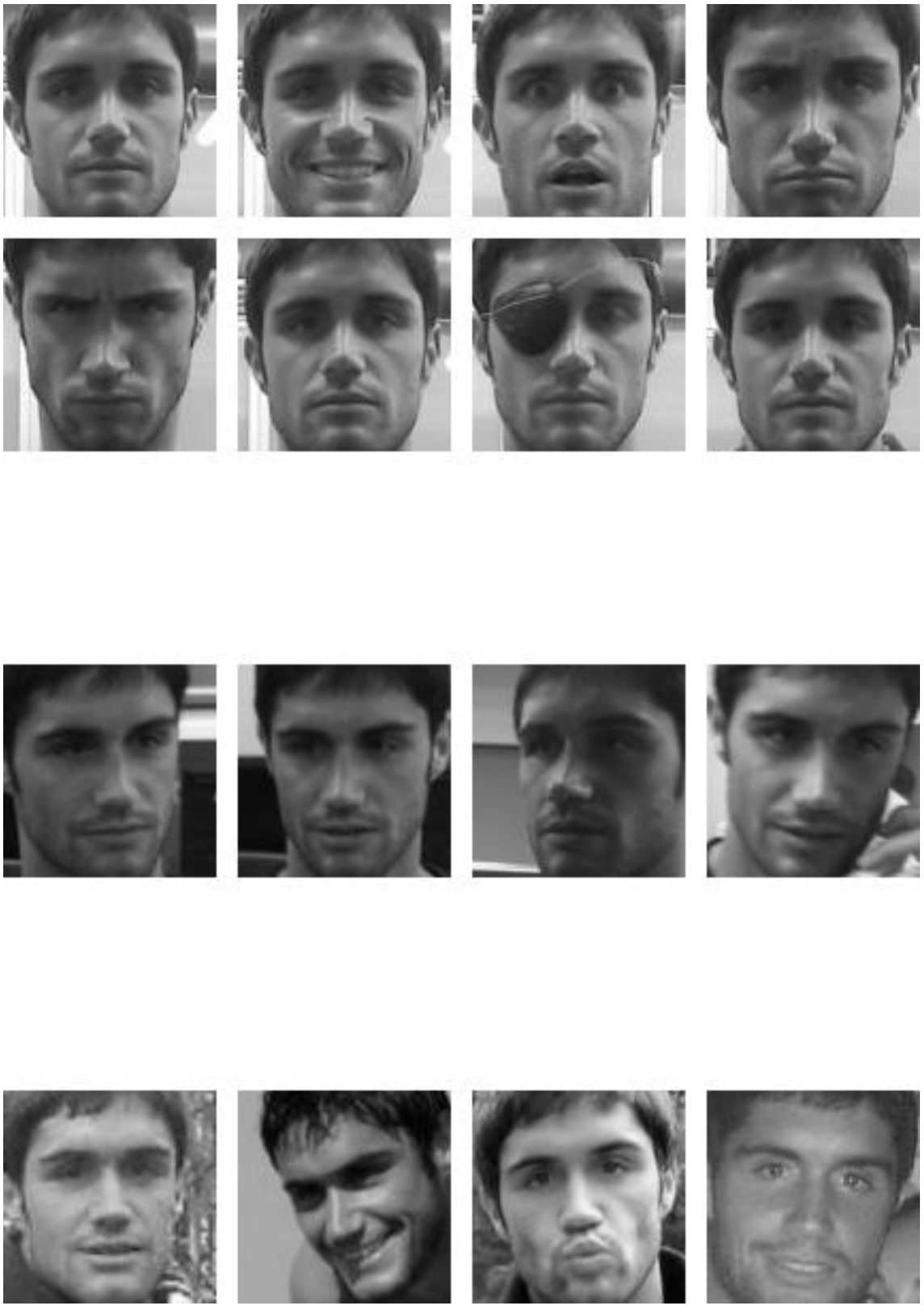

Fig. 6.70 – Conjuntos de imágenes sujeto 70 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.71 – Conjuntos de imágenes sujeto 71 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.72 – Conjuntos de imágenes sujeto 72 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.73 – Conjuntos de imágenes sujeto 73 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

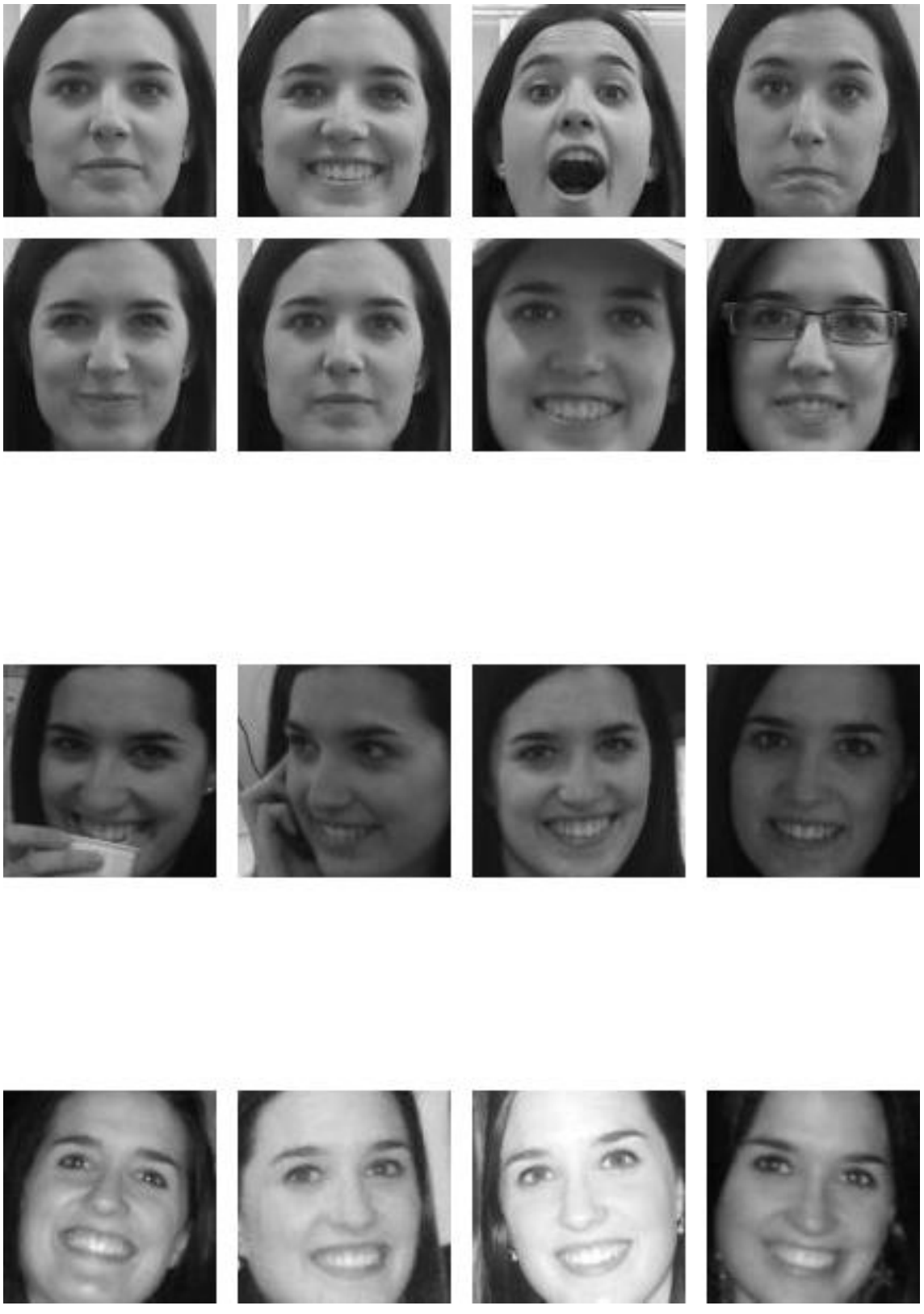

Fig. 6.74 – Conjuntos de imágenes sujeto 74 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

Fig. 6.75 – Conjuntos de imágenes sujeto 75 -

Grupo A: Imágenes tipo Visible .

Grupo B: Imágenes tipo Wild-IPhone.

Grupo C: Imágenes tipo Wild-Libre.

7. CONCLUSIONES

Para la elaboración de este proyecto decidimos comenzar con el análisis de las librerías de imágenes de rostros más utilizadas, pudiendo así saber en que debíamos basarnos para obtener una base de datos fiable y que cubra la mayor parte de las necesidades que puedan surgir.

Para el manejo de todas las librerías así como de las nuestra propia decidimos crear una seria de funciones que facilitasen la interacción y manipulación de las imágenes.

Pese a que pueda parecer la menor de las tareas que se han llevado a cabo para acometer el proyecto, el proceso de recopilación de sujetos y la toma de las imágenes ha supuesto buena parte del esfuerzo que hemos efectuado ya que, pese a lo gratificante, el factor humano siempre es una dificultad.

El producto del trabajo que hemos llevado a cabo es una base de datos completa y variada, que abarca casi la totalidad de los ámbitos que se han tratado previamente en las demás bases de datos y que creemos podrá servir para futuros experimentos en distintas disciplinas.

Hay dos variedades de imágenes cuya importancia destaca por encima del resto, una de ellas, el apartado de imágenes infrarrojas, no ha sido objeto de estudio en este proyecto pero si esta incluida en la librería Mediterranean Faces. El otro tipo de imágenes que diferencia a esta librería de la gran mayoría de imágenes es el que hemos llamado *Wild,* estas imágenes no posadas permitirán poner aprueba el algoritmo en condiciones más parecidas a las que podremos encontrar en la mayoría de aplicaciones de sistemas de reconocimiento facial.

Por último decir que la base de datos será puesta a disposición de todo el que la solicite en la pagina del laboratorio de control de sistemas inteligentes – LCSI – de la Universidad Miguel Hernández de Elche.

[1]. **Wikipedia.** [Sitio web]. http://es.wikipedia.org/wiki/Biometr%C3%ADa

[1] **Vicente Ripoll, Mª Asuncion.** Temario de reconocimiento de patrones. 2012

[2] **Wikipedia.** [Sitio web]. http://en.wikipedia.org/wiki/Viola-Jones_object_detection_framework.

[3] **Wikipedia.** [Sitio web]. http://es.wikipedia.org/wiki/Detecci%C3%B3n_de_caras

[4] **Wikipedia.** [Sitio web]. http://es.wikipedia.org/wiki/Sistema_de_reconocimiento_facial

[5]. **The Ohio State University.** [Sitio web]. http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html

[6] **AT&T Laboratories Cambridge.** [Sitio web]. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[7] **NICTA.** [Sitio web]. http://itee.uq.edu.au/~uqywong6/chokepoint.html

[8] **George Mason University.** [Sitio web]. http://www.itl.nist.gov/iad/humanid/feret/

[9] **Carnegie Mellon University.** [Sitio web]. http://www.ri.cmu.edu/research_project_detail.html?project_id=418&menu_id=261

[10] **Yale University.** [Sitio web]. http://cvc.yale.edu/projects/yalefaces/yalefaces.html

[11] **California Institute of Technology.** [Sitio web]. http://www.vision.caltech.edu/Image_Datasets/faces/README

[12] **Gary B. Huang,Manu Ramesh, Tamara Berg, and Erik Learned-Miller.** Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments.

[13] **Especificaciones técnicas iphone 4.** [Sitio web]. http://www.apple.com/es/iphone/iphone-4/specs.html

[14] **Paul Viola y Michael Jones.** Robust real-time face detection. 2003.

[15] **Ynocente Castro, Mario.** Deteccion de rostros. 2011. http://www.slideshare.net/push4016/deteccion-de-rostros

[16] **Padovani, Andres.** Deteccion y reconocimiento de caras. Tesis de Licenciatura. 2011.